Documentation for OpenKM 7.1

Table of Contents

Table of Contents 2
OpenKM 7.1 44
Hardware and software requirements 46
Requirements 46
Checking the Java version 46
For Windows users 46
Hardware configuration 47
Browser support 48
Operating systems 48
Third-party software requirements 50
Installation 52
Installing on Ubuntu and Debian 54
Preliminaries 54
Check disk size 54
Create user named openkm 54
Check if partners channels are enabled 54
Check server configuration 54
Increase ulimit 54
Checking Java version 55
Install Java 55
Set OpenJDK 1.8 as your default Java version 56
Installing database 56
Change MySQL root password 56
Method 1 56
Method 2 56
Method 3 57
Method 4 57
Check InnoDB is default MySQL engine 57
Create database 57
Installing Tomcat 58
Configuring Tomcat as a service 58
Check the service 59
Installing OpenKM 59
Installing third-party software 59
Start application 60
Check openkm.properties parameters 60
Other common additional configurations setup 61
Installing on RedHat and CentOS 62
Preliminaries 62
Check disk size 62
Create a user named openkm 62
Check server configuration 62
Increase ulimit 62
Checking Java version 63
Install Java 63
Set OpenJDK 1.8 as your default Java version 63
Installing database 63
Change MySQL root password 64
Method 1 64
Method 2 64
Method 3 65
Method 4 65
Check InnoDB is default MySQL engine 65
Create database 65
Installing Tomcat 66
Configuring Tomcat as a service 66
Check the service 67
Installing OpenKM 67
Installing thirdparty software 67

Start application 68
Check openkm.properties parameters 68
Linux Oracle 7.X Throubleshooting 69
Install libreoffice from RPM 69
Install Tesseract 70
Other common additional setup configurations 71
Installing on Windows 72
Preliminaries 72
Checking Java version 72
Install Java 72
Installing database 73
Check InnoDB is default MySQL engine 73
Create database 73
Installing Tomcat 73
Configuring Tomcat as a service 73
Check the service 77
From terminal 77
From Windows service panel 77
From tomcat7w tool 78
Configuring environment variables 79
Installing OpenKM 81
Start application 82
Check openkm.properties parameters 82
Other common additional setup configurations 83
Using the installer 84
Supported databases 85
Other options 86
Using Docker 87
Configuration 88
Docker Compose 89
Installing on a pristine Tomcat 92
Preliminaries 92
Installing Tomcat 92
Copy libraries and configuration files 92
Modify configuration files 92
Installing OpenKM 93
Install third party software 93
Start application 93
Check openkm.properties parameters 93
Database 95
Configure database 95
Configuring MySQL and MariaDB 96
Preliminaries 96
Change default engine 96
Change default charset 96
Database creation 97
Create a database and user 97
Configure your openkm.properties 97
MySQL Timezone 97
Configure Tomcat datasources 98
MySQL: 98
MariaDB: 98
Configure application login 98
Check for JDBC Driver 98
Run application 99
Additional information 99
Remove and create MySQL service in Windows 99
Migrate from Windows to Linux 99
MySQL tuning 101
Sample one 101

Sample two 102
Change MySQL data directory 104
Ubuntu 104
CentOS and RHEL 105
Windows 106
MySQL 8 throubleshooting 107
Could not create connection to database server 107
Strong Password Encryption for Authentication 107
Old JDBC driver 109
The server time zone value is unrecognized 110
Problem connecting to your database 110
Configuring Oracle 112
Database creation 112
Create tablespace 112
Create a database and user 112
Configure your openkm.properties 112
Configure Tomcat datasources 112
Configure application login 113
Run application 113
Configuring PostgreSQL 115
Database creation 115
Create a database and user 115
Configure your openkm.properties 115
Configure Tomcat datasources 115
Configure application login 116
Enable native configuration parameter 116
Run application 116
Additional information 116
PostgreSQL 9.6 and upper 116
Database creation 117
Create a database and user 117
Configure your openkm.properties 118
Configure Tomcat data sources 119
Configure application login 119
Run application 119
Troubleshooting error: 119
You can not connect to MS SQL Server 120
Connection lost after some time 120
Not able to login into OpenKM 120
Not able to connect TCP/ IP to localhostSQL, port 1433 when use instance in the JDBC connection URL 120
Additional information 121
Configuring other databases 122
Configuring Hikari pool 123
Enable debug 123
Backup and restore 124
Detailed steps 125
Backup 125
Restore 125
Backup scripts 126
Additional information 126
Linux backup and restore with BorgBackup 127
Installation 127
Sample script 127
Backup to remote computer with SSH 127
This is an improved SSH version 128
Backup to mapped network 128
Configure crontab 129
Recovery 130
Linux backup and restore with rdiff-backup tool 131

Debian & Ubuntu 131
CentOS & RedHat 131
Preliminaries 131
Sample script 131
Increase the backup period 134
Do remote backup 134
Do remote backup to SMB or CIFS 135
Do PostgreSQL backup 135
Configure crontab 135
Restoring backup 136
Additional information 136
Linux Sample Backup with Duplicity 138
Installation 138
Command line basics 138
Preliminaries 139
Sample script 139
Increase the backup period 142
Perform the remote backup 142
Perform the remote backup to SMB or CIFS 142
Do the PostgreSQL backup 143
Do the remote backup to Amazon s3 143
Configure crontab 144
Troubleshooting deleting old backups 144
Restoring backup 144
Additional information 145
Linux backup and restore with rsync 146
Preliminaries 146
Sample script 146
Do backup with rotation 148
Do remote backup 149
Do incremental backup with backup-dir 149
Do incremental backup with link-dest 150
Do remote backup to SMB or CIFS 150
Do PostgreSQL backup 150
Configure crontab 151
Rsync and FTP 151
Restoring backup 152
Additional information 152
Linux backup with LFTP 153
Preliminaries 153
Sample script 153
Configure crontab 155
Cluster configuration 156
Proxy node 158
Master node 159
Slave nodes 160
Sharding configuration 162
Sharding and Docker 162
Instance replication 165
Different implementations 166
Sample Linux implementation 166
How to move to another server 168
Linux 168
Preliminaries 168
Stop OpenKM service 168
Backup the database 168
Restore the database 168
Copy binary files 169

Configure service in the new server 169
Install third-party tools 169
Start OpenKM service 170
Additional steps 170
Moving license 170
Final checks 170
Setup 171
Antivirus configuration 172
Linux 172
Debian / Ubuntu 172
Red Hat / CentOS 172
Windows 173
Clamd Service Setup 174
Freshclam Virus Signature Auto Update 175
Cleaning older activity log data 177
MySQL 177
Linux 177
Configure crontab 177
Windows 178
Database replication with federated tables 178
Configuring Apache HTTP Reverse-Proxy 180
Internal IP vs external IP 180
Debian and Ubuntu 181
Check the configuration 183
Red Hat and CentOS 183
Check the configuration 184
Windows 184
Max OS X 185
Check the configuration 185
Troubleshooting 186
Additional information 186
Configuring Apache HTTP for changing context URL 187
Additional information 187
Configuring Apache HTTPS Reverse-Proxy 188
Internal IP vs external IP 188
Debian and Ubuntu 189
Redirect all HTTP connections to HTTPS 191
Check the configuration 191
Red Hat and CentOS 191
Check the configuration 192
Troubleshooting 193
Additional information 193
Configuring Nginx HTTP Reverse-Proxy 194
Internal IP vs external IP 194
Debian and Ubuntu 195
Check the configuration 196
Red Hat and CentOS 196
Check the configuration 197
Configuring Lucene Analyzer 199
Configure an Analyzer 200
Configuring OCR engine 201
Tesseract 201
Installation 201
Check installation 202
Configuration 202
Additional information 203
Cuneiform 203
Installation 203
Check installation 204

Configuration 204
Abby OCR for linux 204
Installation 205
Check installation 205
Configuration 205
Dictionary 206
Configuration 206
OCR rotate configuration 206
Configuration 206
Additional information: 206
Configuring OpenSSL 208
Create OpenSSL configuration file 208
Create Certification Authority (CA) 209
Create Apache Certification 210
Apache configuration 210
Certification revocation 211
Add CRL to Apache 212
Configuring SWFTools 213
Ubuntu 213
Red Hat and CentOS 213
Improving performance 214
Debian 214
Improving performance 215
Windows 215
Improving performance 215
Troubleshooting 216
Conversion problems with big files 216
Segmentation fault (core dumped) 216
Error with PDF with copy protected 216
Configuring mail 218
Change mail configuration 218
Gmail example 218
Office 365 218
Email server with old SSL configuration 219
Additional information 219
Testing configuration 219
Usage 219
Configuring Tomcat memory utilization 221
OpenJDK 1.8 221
OpenJDK 1.7 221
Configuring Tomcat port 222
Change tomcat ports 222
Configuring Tomcat session time-out 224
Configuring Tomcat SSL/TLS 225
Self-signed Certificate 225
Create the Certificate Key store 225
Edit the Tomcat Configuration file 226
Check configuration 226
CA Certificate 226
Prepare the Certificate Key store 226
Create the Certificate Signing Request 226
Install the Root Certificate 227
Install the Intermediate Certificate 227
Install the Primary Certificate 227
Edit the Tomcat Configuration file 227
Check configuration 228
CA Certificate without preparing keystore 228
Obtain p12 certficate 228
Install p12 Certificate 228
Change alias to Primary Certificate 228

Let's Encrypt 228
Client installation 228
Certificate generation 229
Tomcat configuration 229
Other configuration 229
Known issues 230
Windows uses a different protocol 230
ca-bundles certificates 230
Additional information 230
Configuring Tomcat timezone 231
Configuring openkm.xml 232
Enable external openkm.xml 232
Memory sample 232
Database authentication sample 233
Ldap Authentication sample 233
Multiple providers sample 234
Configuring jPBM 236
Configuring safe trash purge 237
Configuring Tomcat to work behing an Internet proxy 238
Linux 238
Windows 239
Step one 239
Step two 240
Change repository home 242
Understanding the contents of repository folders 242
Move entire repository location 243
Sample1 243
Windows 243
Linux 243
Move repository folders individually 243
Sample 244
Windows 244
Linux 244
Troubleshooting 244
Windows mapped network disk is not working 244
Change default role names 245
Change the default roles 245
Sample of appContext.xml file 246
Tools 248
Enable extensions 250
Enable WebDAV 252
Linux 252
Windows 253
WebDrive 253
Windows Web Folders 254
Windows XP's "My Network Places" 255
Other tools 255
Mac 255
Mac OS X finder. 255
Troubleshooting 256
Slow WedDAV performance on Windows 256
Turn off file locking 256
LDAP configuration 257
OpenLDAP 257
Active Directory 257
Uncategorised examples shared by users 257
Tools 257
OpenLDAP example with URL base at login 258

LDAP Structure 258
Step 1 - configuration parameters 259
Step 2 - configure login 260
Another option to configure login 261
Active Directory example with referral enabled 263
LDAP Structure 263
Step 1 - configuration parameters 264
Step 2 - configure login 265
Another option to configure login 266
Active Directory example with login based on filtering users by roles 268
LDAP structure 268
Step 1 - configuration parameters 269
Step 2 - configure login 271
Another option to configure login 271
Active Directory basic configuration 274
LDAP structure 274
Step 1 - configuration parameters 274
Step 2 - configure login 276
Another option to configure login 277
Active Directory advanced configuration 278
LDAP structure 278
Step 1 - configuration parameters 278
Step 2 - configure login 280
Another option to configure login 281
Active Directory mixed Configuration 282
LDAP structure 282
Step 1 - configuration parameters 282
Step 1 - configure login 284
Active Directory example with remember-me services 286
LDAP structure 286
Step 1 - configuration parameters 287
Step 2 - configure login 289
Another option to configure login 290
LDAP best practices for filtering users and roles 292
Example based on single LDAP group 292
Example based on ROLE_USER and ROLE_ADMIN group 292
ClearOS LDAP configuration 294
LDAP Structure 294
Step 1 - configuration parameters 294
Step 2 - create a Certificate 296
Step 3 - configure login 297
Another option to configure login 298
LDAP troubleshooting 300
No UserDetailsService registered error 300
javax.naming.PartialResultException error 300
Can't connect to LDAP server 300
Slow login or not able to login 301
Force all users to be members of ROLE_USER 301
org.springframework.beans.factory.parsing.BeanDefinitionParsingException 302
LDAP: error code 8 - BindSimple: Transport encryption required 303
Ldap error: Caused by java.security.cert.CertificateException: No subject alternative DNS name matching
Log Configuration 305304
Logback based Log Configuration 305
Third-party application execution 305
Enable log authentication 305

Log4j based Log Configuration 305
Third-party application execution 306
Enable log authentication 306
OpenOffice and LibreOffice configuration 307
Linux 307
Debian & Ubuntu 307
CentOS 307
Windows 308
Check configuration 309
Security configuration 310
Diferences betweem simple and recursive 311
Extending security 312
Configuration for security changes which take a long time 312
SSO Configuration 314
Active Directory 314
Linux Machine 315
Kerberos configuration file 316
Configuration files 316
Configure LDAP in OpenKM 319
Client browsers: 319
Throubleshootings 319
Enable log for SSO authentication 320
Check kerberos authentication from linux command line 320
Listing keys 320
Configuration 322
Register parameters 322
Configure parameters 322
Register metadata group 323
Crontab task 324
Register reports 324
Configuring Two Factor Authentication 326
Prerequisite 326
OpenKM configuration 326
Scan QR code 326
Google Authenticator use 327
Configuring CIFS 328
Database Authentication 328
OpenKM server 328
OpenKM Configuration 329
Windows client 330
Active Directory Authentication 331
Active Directory 331
OpenKM server in Active Directory configuration 332
Introduce server in domain 332
Redirect the required ports 332
kerberos configuration file 333
cifs.login.config File 334
Enable Java login configuration. 334
OpenKM Configuration in Active Directory Configuration 335
Windows client in Active Directory configuration 337
Cache configuration 338
Configuring CADViewer 339
Configuring VidSigner 342
Configuring search highlighter 344
Configuring email digital signature 345
More info 345
Configuring anonymous user 346
Throubleshooting 347

Server configuration 348
Packet for query is too large (MySQL) 348
Not able to connect from other computer in the network 348
Error downloading document as PDF 348
IMAP SSL certificate error 348
OpenKM server has no direct Internet connection 349
Access denied installing OpenKM as service on Windows 7 349
Problems with character encodings 349
PermGen errors 349
LDAP Sizelimit exceeded 350
Behind IIS proxy you get error 404 or 404.15 350
IIS gets IOException: Internal Server Error while uploading file 352
Solution one 352
Solution two 352
Zoho fails with SSL connection 353
CMIS bad URL behind Nginx proxy 353
Sigar java.lang.UnsatisfiedLinkError Known issue in Windows 354
WARN net.sf.ehcache.hibernate.strategy.AbstractReadWriteEhcacheAccessStrategy 354
OpenOffice is not starting on Windows 354
RedHat and CentOS issues 356
CentOS 8: OpenKM not starting as a service automatically 356
ImageMagick in CentOS 8 357
Client configuration 359
ZIP filenames with wrong charset 359
Microsoft Office Addin disappears after Office restart 360
Error in Uploader / Scanner applet 360
Browser configuration 361
Firefox cause error when click on "Copy to clipboard" 361
Unable to run Scanner applet under Linux 361
Internet Explorer error "Document name is different" 361
Internet Explorer error downloading files in HTTPS / SSL connections 361
Google Chrome and Java applets 362
Best practices reporting issues 363
Getting the stack trace 363
Linux 363
Windows 363
Sample of stack trace capture 364
Administration guide 365
Configuration parameters 369
Edit a parameter 369
Delete a parameter 369
Backup configuration parameters 369
Restore configuration parameters 370
Check configuration 370
Multitenant configuration 370
Recommended configuration parameters 371
Preview parameters 374
Linux 374
Windows 375
LDAP configuration parameters 377
Sample query 377
LDAP queries 377
Parameters 378
Sample configuration 380

User interface configuration parameters 383
Workflow configuration parameters 388
Notification configuration parameters 390
Substitutions 393
Document version numeration configuration 394
Version numeration schemas 394
Migration between versions 395
Password validation configuration 396
Sample configuration 397
Switch application to read-only mode 398
Security configuration parameters 399
Extensions configuration parameters 404
HTML Editor 4 404
HTML Email editor 406
HTML task manager 407
HTML Editor 3 408
DXF applet viewer 413
Extratab extension 414
External preview extension 414
External tab document extension 414
External tab folder extension 415
External tab mail extension 415
External tab record extension 415
Menu options extension 416
Performance configuration parameters 417
Text extractor queue 417
OpenOffice LibreOffice conversion service 418
Download files and folders as ZIP file 419
Antivirus checker 419
Re-indexing the whole Lucene repository 421
Execution timeout 421
Upload bandwidth 422
Remote conversion service 422
Activity log actions 422
Dashboard activity 423
Repository stats 423
Lucene 424
OpenOffice & LibreOffice configuration parameters 425
WebDAV configuration parameters 426
CSV configuration parameters 427
Antivirus configuration parameters 428
Other configuration parameters 430
Experimental configuration parameters 439
AutoCAD configuration parameters 442
Thesaurus configuration 443
FTP configuration parameters 445
Two factor authentication configuration parameters 446
Multi-tenant configuration parameters 447
File plan 448
Enable the File plan 448
Mandatory 448

Optional 448
How to construct the File plan 448
MIME types 451
Add new MIME type 451
Edit MIME Type 452
Delete MIME type 452
Backup MIME types 452
Restore MIME types 452
Statistics 453
Check the status of the Pending task queue 453
Check the status of the Text extraction queue 453
Check the status of the Antivirus check queue 454
Metadata 455
Edit metadata 456
Metadata XML definition 457
Basic XML skeleton 457
Metadata group definition 457
Basic example 458
Using file system DTD 459
How metadata is stored into OpenKM 459
Metadata fields 462
Metadata Checkbox field 464
Checkbox 464
Metadata Iframe field 466
Metadata Select field 468
Basic select example 470
Allow multiple choices 470
Hierarchical relationship 470
Based on SQL query 471
Based on className 472
Suggested options 472
Metadata Separator field 473
Metadata Suggestbox field 474
Based on filterQuery and valueQuery 475
Based on className 476
Metadata Text field 477
Metadata TextArea field 478
Metadata Validator element 480
Basic example 481
Regular expression 481
Metadata Option element 482
Basic select with selected option 482
Hierarchical relationship 483
Metadata Input field 484
Metadata VerticalPanel field 486
Metadata HorizontalPanel field 487
Manage users and roles 488
Manage users 489
Create a user 490
Edit a user 490
Delete a user 490

Change the user profile 490
Export the users 491
Enable twitter account 491
Manage user mail accounts 492
Create a new mail account 493
Edit a mail account 494
Delete a mail account 494
Browser emails in the mail server 494
Manage mail account filters 494
Create a mail account filter 496
Edit a mail account filter 496
Delete a mail account filter 496
Create a mail account rule 496
Edit a mail account rule 496
Delete a mail account rule 497
Manage roles 498
Create a role 498
Edit a role 498
Delete a role 499
Logged users 500
Real time user activity log 500
Message queue 501
Send a message 502
Edit a message 502
Delete a message 502
Delete all messages queue 502
Refresh message queue 502
Profiles 503
Create a new profile 503
Edit a profile 503
Delete a profile 503
Clone a profile 504
Profile General tab 505
General section 505
Misc section 505
Chat section 507
Wizard section 508
Pagination section 508
Section General 510
Profile Components tab 512
Stacks section 512
Dashboard section 512
Tab section 513
Tab document section 514
Tab folder section 514
Tab mail section 515
Tab record section 515
Tab search section 516
Tab basic section 517
Tab advanced section 517
Profile Menu tab 519
Menu section 519
Menu bookmark section 519
Menu tool section 520

Menu help section 521
Menu file section 521
Menu edit section 523
Profile Notification tab 525
Events 525
Common 525
Document 525
Folder 527
Mail 527
Record 528
Profile Other tab 530
Toolbar section 530
File browser section 532
Relation types 533
Types 533
Bidirectional 533
Parent - child 533
Many-to-many 534
Create a relation 534
Edit a relation 534
Delete a relation 534
CSS 535
Create a new CSS 535
Edit a CSS 535
Delete a CSS 535
Clone a CSS 535
Download a CSS 536
Samples 536
Large font sample 536
Medium font sample 536
Default font sample 536
Reports 538
Create a new report 538
Edit a report 538
Delete a report 539
Show report report parameters 539
Execute a report 539
Reports pack 540
Template reports 540
Authentication 540
Documents 540
Folders 543
Mails 544
Others 545
Activity log 546
Auth actions 546
Document actions 547
Folder actions 549
Mail actions 550
Record actions 551
Property group actions 552
Node actions 552
Repository actions 553
Admin actions 553
Misc actions 560
Workflow 561

Register workflow 561
Workflow forms definition 563
Basic XML skeleton 563
Workflow task form definition 563
Basic example 564
Using file system DTD 565
Workflow fields 566
Workflow Button field 569
Workflow Download field 571
Basic download example 571
Dynamic download example 572
Workflow Validator element 573
Basic example 574
Regular expression 575
Workflow Node element 576
Basic download example 576
Workflow Upload field 577
Basic upload example 578
Document update 578
Upload example with dynamic parameters 578
Workflow Print field 580
Basic print example 580
Dynamic print example 581
Workflow Checkbox field 582
Checkbox 582
Workflow Input field 583
Workflow Select field 585
Basic select example 586
Allow multiple choice 586
Hierarchical relationship 587
Based on SQL query 587
Based on className 588
Workflow Separator field 589
Workflow Suggestbox field 590
Based on filterQuery and valueQuery (without table attribute) 591
Based on className 591
Workflow Text field 593
Workflow TextArea field 594
Workflow Option element 595
Basic select with selected option 595
Hierarchical relationship 596
Workflow VerticalPanel field 597
Workflow HorizontalPanel field 598
Automation 599
Preliminar concepts. 599
Understanding Validators and Actions 601
Creating a rule 602
Editing a rule 602
Edit general parameters 602
Edit action and validation definitions 602
Deleting a rule 603

Built-in Automation Validator plugins 604
Built-in Automation Action plugins 607
Stamp 614
Create new stamp based on text 614
Create new stamp based on image 616
Edit stamp 618
Delete stamp 618
Test stamp 619
Crontab 620
Create a new crontab job 622
Edit a crontab job 622
Execute a crontab job 622
Delete a crontab job 622
Built in crontab tasks 623
OMR templates 625
Add a new OMR template 625
Preliminars 626
Create a new template 627
Update asc training files 627
Update .fields training files 630
Check the template 632
OCR templates 634
Validation Control field description 634
Data Capture field description 636
Understanding OCR Validation Control and OCR Data Capture 637
Create a new OCR template 638
Edit a OCR template 640
Edit a OCR template fields 640
Edit a OCR Validation Control 640
Edit a OCR Data Capture 640
Delete a OCR template fields 641
Delete a OCR Validation Control 641
Delete a OCR Data Capture 641
Check OCR template 641
Check recognise a document 641
Delete a OCR template 642
Enable OCR Data Capture 642
Built in OCR Validation Control plugins 643
Built in OCR Data Capture plugins 645
Languages 647
Create a new language 647
Edit a language 647
Delete a language 647
Edit language translations 648
Export a language translations 648
Import a language translations 648
Language translations 649
How to create a new translation 652
A sample of new language base on Oromo from Ethiopia 652
Utilities 653
Admin permissions 655
Change user level access 655
Cache stats 656
Clean all cache 657

Refresh cache stats 657
Enable cache stats audit trail 657
Disable cache stats audit trail 657
Clean cache stats audit trail 658
Check email 659
Check email 659
Check text extraction 660
Check text extraction 660
Clean conversion cache 661
Clean conversion cache 661
Database query 663
Execute JDBC query 663
Execute Hibernate query 664
Allowed sentences 664
Download tokens 665
Delete token 665
Execute JAR 666
Folder style 667
Create a new Folder style 667
Clone a Folder style. 668
Edit a Folder style 668
Delete a Folder style. 668
Create support zip 669
Hibernate stats 670
Information provided by Hibernate stats 670
Enable Hibernate stats 670
Clear Hibernate stats 671
Disable Hibernate stats 671
LDAP Search 672
List indexes 674
Search indexes 675
Enable show terms 676
Disable show terms 676
LogCat 677
Purge all logs 677
Download file as ZIP 677
Browser log file 677
Plugins 679
Profiling stats 682
Enable profiling stats 683
Disable profiling stats 683
Clear profiling stats 683
Refresh profiling stats 683
List of detailed calls 683
Purge permissions 684
Purge 684
Purge trashes 685
Purge 685
Rebuild indexes 686
Rebuild 686
Repository checker 687
Check repository 687

Repository export 688
Export 688
Understanding output format 688
Case metadata and history are not checked 688
Case only metadata is checked 689
Case only history is checked 690
Case metadata and history are checked 690
Repository import 691
Import 692
Scripting 693
Execute script as part of application starting process 693
Execute script as part of application stopping process 693
Scripting in Automation 693
System properties 695
Thesaurus 697
Build thesaurus 697
User guide 698
Menu bar 699
Quick search box 699
ToolBar 699
Desktop 699
Navigation 699
Browser 700
Properties 700
Search 701
Dashboard 702
Administration 703
Status 704
Quick start 705
Loggin in 705
Create a taxonomy 705
Add documents 707
Edit documents 708
Search documents 710
Essential fundamentals 712
Basic concepts 712
Understanding taxonomy, keywords, categories and metadata groups. 712
Taxonomy 713
Keywords 713
Categories 714
Metadata 714
Plan your data organization 715
Good practice 716
Working with the menu bar 717
File 717
Edit 720
Tools 723
Bookmark 728
Templates 729
Help 729
Working with the toolbar 730
Working with desktop 735
Navigation 735
Browser 738
Pagination 742

Properties 744
Tabs on nodes of type document 744
Tabs on nodes of type folder 747
Tabs on nodes of type mail 750
Tabs on nodes of type record 753
Shortcuts 756
Working with search 758
Searches panel 758
Search panel 759
Operators and wildcards 759
Default search 759
Advanced search 760
Control panel 763
Results 764
Working with Lucene queries 767
Term modifiers 767
Wildcard searches 767
Proximity searches 767
Range searches 768
Boolean operators 768
Working with Lucene queries 768
Term modifiers 769
Wildcard searches 769
Proximity searches 769
Range searches 769
Boolean operators 769
AND 770
+ 770
NOT 770
- 770
Grouping 771
Escaping special characters 771
AND 771
+ 771
NOT 771
- 772
Grouping 772
Escaping special characters 772
Working with metadata fields 772
Working with the dashboard 773
User view 773
General view 774
News 775
Mail 775
Toolbar 775
Mail browser 776
Preview 776
Mail accounts 777
Workflow 777
Workflow lists 777
Workflow detail 777
Keyword map 778
Toolbar 778
Browser 778
Keywords lists 779
Results controller 779
Tasks 780
Toolbar 780
Filter 781
Browser 781

Calendar 783
Messaging 783
Message type 783
Browser 784
Preview 784
Forum 784
Wiki 785
Working in status bar 786
OpenKM news alert 786
The connected user 786
User repository size 786
Chat 786
Alerts 786
HTML documents on edition 787
Message panel 787
User email signature 789
Bulk Uploader 790
Usage 790
Live Edit 791
Use 791
HTML editor extension 793
Features 793
Edit HTML 794
Editor panel 794
Preview panel 795
AutoCAD preview extension 796
Toolbar 796
Default preview 799
Cryptography extension 801
Digital Signature extension 802
Sign document in repository 802
Upload and sign new document 803
Digital Signature viewer extension 804
Verify a signed document 804
Document expiration extension 805
Add document expiration 805
Extratab extension 806
Configuration 806
Sample 806
External preview extension 808
Configuration 808
Sample 808
Configuration 808
Usage 809
External tab document extension 810
Configuration 810
Sample 810
Configuration 810
Usage 811
External tab folder extension 813
Configuration 813
Sample 813
Configuration 813
Usage 814

External tab mail extension 816
Configuration 816
Sample 816
Configuration 816
Usage 817
External tab record extension 819
Configuration 819
Sample 819
Configuration 819
Usage 820
Menu options extension 822
Configuration 822
Usage 822
Forum extension 824
Editor features 824
Create new topic 825
Create reply 825
Update post 826
Delete post 826
Create forum 826
Update forum 826
Delete forum 826
Stamp extension 827
Document stamping 827
Using OpenKM from external applications 828
Microsoft Office Add-in 828
Desktop sync 828
Digital Signature Client 828
Scanner client 828
Desktop Importer tool 828
Microsoft Office Add-in 830
Configuring add-in 830
Editing documents from Microsoft Office 830
Importing mails from Microsoft Outlook 831
Add new translation 831
Know issues 832
Digital Signature Client 833
Configuring 833
Sign documents 833
Scanner client 834
Requirements 834
Configuring 834
Compression CCITT Group4 835
Setting destination folder 835
Setting scanner configuration 835
Scanning 835
Convert scanned images to PDF 835
Uploading 836
Add images to an existing PDF file 836
Remove pages from existing PDF file 837
Desktop Sync 838
Limitations of the application 840
Configuring 840
Change initial configuration 841

Synchronizing 841
Remove synchronized node 842
Resolving conflict 842
Desktop importer tool 844
Images 844
Working with PDF tools 845
Steps to enable PDF tools in the classic user interface 845
Migration guide 846
Troubleshooting 846
Changelog 848
OpenKM 7.1 Professional 848
v7.1.20 848
v7.1.19 848
v7.1.18 850
v7.1.17 850
v7.1.16 851
v7.1.15 851
v7.1.14 852
v7.1.13 852
v7.1.12 853
v7.1.11 853
v7.1.10 854
v7.1.9 854
v7.1.8 854
v7.1.7 854
v7.1.6 855
v7.1.5 855
Updater utility 856
Upgrade stages 856
Manual mode 857
Updater utility english video 859
Updater utility spanish video 860
Tomcat update 861
Linux 861
Preliminaries 861
Tomcat replacement 861
Windows 863
Preliminaries 863
Tomcat replacement 863
Manual update 867
Cluster update 868
Master node 868
Slave nodes 868
Migrating from 7.1.3 to 7.1.4 870
Migration 870
Before the update 870
Update process 870
After the update 871
Step 1 - Database changes 871
MySQL 871
Step 2 - Execute the script 871
Step 3 - Final database changes 872
MySQL 872
Step 3 - Reindex the whole repository 872
Changes from version 6.4 873
Changes 873
Deprecated features from version 6.4 873
Development 874

OpenKM professional portable development environment 874
Javadoc 875
Personal development environment 876
Requirements 876
How to get the personal development environment 876
How to install 877
License activation 877
How to upgrade the development environment 877
Configuring IDE from a clean Eclipse 878
Requirements 878
Software: 878
Hardware: 878
Download eclipse 878
Configure Java 878
Configure tomcat server 880
Maven support for Eclipse 886
Subversion support for Eclipse 886
Configuring Maven in your OS 888
Maven tips 888
Memory problems 888
Configure eclipse to use local Maven 889
Eclipse tips 890
Debug tomcat 890
Debug remote tomcat server 890
Eclipse configuration 890
Debug GWT frontend user interface with GWT Super Dev Mode 892
Enable source for debugging 892
Eclipse configuration 893
Debug GWT frontend user interface with GWT Shell 902
Enable source for debugging 902
Eclipse configuration 903
Eclipse know troubleshooting 910
Maven raise error 206 the filename or extension is too long 910
Development tips 911
Tomcat application deployment 911
Compile only GWT frontend module 911
Configure GWT build from eclipse 911
Compile only one GWT frontend permutation 913
Enable only firefox permutation 913
Enable only chrome permutation 914
Enable only IE permutation 914
Configure number of CPU's to be used for compiling GWT frontend 914
Deploying script sample 915
Profiling application 916
Edit Tomcat startup scripts 916
Authentication 916
Monitoring Remote JVM Over SSH 917
HPROF 918
Application version compatibility table 919
Change default login screen 920
Configuring Tomcat external resource 920
Change default login web page 920
Enable frontend top banner section 921
Change folder style by metadata value 923
Configure sample 923

Create a new Metadata group 923
Create new Folder styles 923
Create a new Automation task 924
Check configuration 926
Creating a JAR file with Eclipse 930
Images 930
Sample JAR file 933
Creating plugins 934
Register a new plugin 935
Creating your own Antivirus Analyzer plugin 936
Method description 936
Example of the clamav antivirus analyzer implementation 937
Creating your own Automation Action 938
Methods description 939
Understanding env variable 940
Understanding params variable 940
Description of available ParamType values: 941
Description of available Src values: 941
Detecting recursion 942
Example 942
Creating your own Automation Validation 945
Methods description 946
Understanding env variable 947
Understanding params variable 947
Description of available ParamType values: 948
Description of available Src values: 948
Example 949
Creating your own Crontab plugin 952
Method description 952
Example of the disposition cron implementation 953
Understanding the crontab job execution context 954
Creating your own Default Form Values 955
Method description 955
Example of the default form values implementation 955
Metadata definition 956
Creating your own Document Converter 957
Example of the Document converter implementation 958
Creating your own Form Validator plugin 960
Methods description 960
Example of the Form validator implementation 961
Creating your own NodePaginator plugin 963
Method description 964
Example 964
Creating your own NodeProperties plugin (customize search results objects) 968
Method description 969
Example of NodeProperties implementation 969
Creating your own OCR Validation Control plugin 971
Methods description 971
Example: 972
Creating your own OCR Data Capture plugin 974
Methods description 974
Example 975
Creating your own Option Select Values plugin 977
Methods description 977

Example of the Option select values implementation 978
Creating your own Principal Adapter 979
Methods description 980
Example of the Principal adapter implementation 982
Creating your own Rest Plugin (extending REST API) 985
Method description 985
Example of REST plugin implementation 986
Example of REST plugin implementation what returns file 986
Objects declaration 987
String scenario 987
Complex Object scenario 988
Example of complex classes: 988
Creating your own Suggestion Analyzer 990
Methods description 990
Example of Suggestion Analyzer 991
Creating your own Suggestbox plugin 993
Methods description 993
Example of implementation 994
Creating your own Stack Metadata Adapter 996
Methods description 997
Example of the Stack Metadata implementation 998
Creating your own Text extractor 1006
Methods description 1006
Example of the Text extractor implementation 1007
Creating your own Version Number Adapter 1008
Methods description 1009
Example of the Version number adapter implementation 1010
Creating your own security access manager 1011
Methods description 1012
Example of Security access manager implementation 1012
Creating templates 1015
Creating OpenOffice and LibreOffice templates 1016
Preliminaries 1016
Creating the template 1016
Date format 1017
Dynamic tables 1018
Creating PDF template 1019
Preliminaries 1019
Creating the template 1020
Date format 1021
Creating Text templates 1022
Preliminaries 1022
Creating the template 1022
Date format 1023
Creating your own login module 1024
PrincipalAdapter sample based on properties 1026
Creating your own parameters 1030
Sample 1030
Creating your own password validator 1031
Sample 1032
Creating your own reports 1034
Introducing to reports based on SQL 1035

Create datasource 1035
Testing SQL sample report 1036
Closer look into JRXML file 1039
Query section 1039
Fields section 1040
Remote image section 1040
Using fields in report 1040
Introducing to reports based on Java 1041
Testing Java sample report 1041
Understanding the code sample 1043
Using in report 1044
Introducing to reports with filtering 1046
Understanding filtering 1046
Params XML file 1046
Using fields in report 1046
Java sample 1047
SQL sample 1049
Params form fields 1051
Basic XML skeleton 1051
Basic example 1051
Using file system DTD 1052
Params fields 1053
Params Validator element 1055
Basic example 1056
Regular expression 1056
Params Input field 1057
Params Suggestbox field 1058
Based on table, filterQuery and valueQuery 1059
Based on filterQuery and valueQuery (without table attribute) 1060
Based on className 1061
Params Checkbox field 1062
Checkbox 1062
Params TextArea field 1063
Params Select field 1064
Basic select example 1065
Allow multiple choice 1065
Hierarchical relationship 1065
Based on SQL query 1066
Based on className 1067
Suggested options 1067
Params Separator field 1068
Params Text field 1069
Creating your own thesaurus 1070
Database stuff 1073
Webservices 1074
CMIS 1075
CMIS clients 1075
Supported operations 1076
Create a session 1078
Browser binding 1078
Atom binding 1078
Web services 1079
Retrieve object samples 1081

Get root folder 1081
Get object by ID 1081
Get object by path 1082
Get folder children 1082
Create object samples 1084
Create folder 1084
Create document 1085
Copy a document to a folder 1085
Update object samples 1087
Update object properties 1087
Rename object 1087
Move object samples 1089
Move an object from a folder to a new folder 1089
Delete object samples 1090
Delete object 1090
Unsupported and partially supported methods 1091
Partially supported 1091
CMIS CmisObject class 1091
Unsupported 1091
CMIS Session class 1091
CMIS Document class 1091
CmisSync 1092
Requirements 1092
Configuration 1092
Poll Interval 1092
Contextual Menu 1093
Tomcat performance 1094
RESTful 1095
Sample usage 1095
Folder samples 1097
Document samples 1097
Search samples 1098
Security samples 1098
Metadata samples 1099
Notes samples 1099
Conversion samples 1099
Swagger 1101
How to accessing Swagger 1101
Playing with Swagger 1103
RestClient sample 1108
Login 1108
Get roles 1109
Extending frontend UI 1111
Enable sample extensions 1111
Frontend UI hello world sample 1113
Description 1113
Sample 1113
Create a new GWT module 1113
Add module into OpenKM frontend UI 1113
Create new widget 1113
Use widget into OpenKM 1115
Results 1115
Frontend UI extensions 1116
Widget library 1118

Tab workspace 1118
Tab Document 1118
Tab folder 1119
Main menu 1120
Toolbar button 1120
Dashboard toolbar button 1120
TabWorkspace Extension 1122
Methods 1122
Basic sample 1122
Sample with iframe 1122
WorkspaceHandlerExtension 1125
Methods 1125
Sample 1125
WorkspaceComunicator 1126
Methods 1126
Sample 1126
TabDocument Extension 1127
Methods 1127
Basic sample 1127
DocumentHandlerExtension 1128
Methods 1128
Sample 1128
TabDocumentComunicator 1130
Methods 1130
Sample 1131
TabFolderExtension 1132
Methods 1132
Sample 1132
FolderHandlerExtension 1134
Methods 1134
Sample 1134
TabFolder Comunicator 1136
Methods 1136
Sample 1137
TabMail Extension 1138
Methods 1138
Sample 1138
MailHandlerExtension 1140
Methods 1140
Sample 1140
TabMail Comunicator 1142
Methods 1142
Sample 1142
TabRecord Extension 1143
Methods 1143
Sample 1143
RecordHandlerExtension 1145
Methods 1145
Sample 1145
TabRecord Comunicator 1147
Methods 1147
Sample 1147

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ToolBarBox Extension 1148
Methods 1148
Sample 1148
DashboardHandlerExtension 1149
Methods 1149
Sample 1149
ToolBarComunicator 1151
Methods 1151
Sample 1151
ToolBarButton Extension 1152
Methods 1152
Sample 1152
ToolBarHandler Extension 1154
Methods 1154
Sample 1154
MenuBarExtension 1156
Contructor 1156
Sample 1156
MenuItemExtension 1158
Constructors 1158
Sample 1158
General HandlerExtension stuff 1160
DashboardHandlerExtension 1160
Methods 1160
Sample 1160
LanguageHandlerExtension 1160
Methods 1160
Sample 1160
NavigatorHandlerExtension 1161
Methods 1161
Sample 1161
PropertyGroupHandlerExtension 1161
Methods 1161
Sample 1162
WidgetHandlerExtension 1162
Methods 1162
Sample 1162
WorkspaceUserPropertiesHandlerExtension 1163
Methods 1163
Sample 1163
General Comunicator stuff 1164
GeneralComunicator 1164
Methods 1164
Sample 1167
DashboardComunicator 1167
Methods 1167
Sample 1168
FileBrowserComunicator 1168
Methods 1168
Sample 1169
NavigatorComunicator 1169
Methods 1169
Sample 1171
SearchComunicator 1171
Methods 1171
Sample 1172
UtilComunicator 1172
Methods 1172

Sample 1174
UI constants 1175
UIMenuConstants 1175
Sample 1175
UIDashboardConstants 1175
UIDesktopConstants 1176
Sample 1176
UIDialogConstants 1176
UIDockPanelConstants 1176
Sample 1176
UIFileUploadConstants 1176
UISearchConstants 1176
Frontend UI extensions stuff 1177
Internatinalization 1177
Sample 1177
Registering RPC (Remote Process Call) service 1178
Complete frontend extension sample 1180
Description 1180
Installation 1181
Implementation details 1182
index.jsp 1182
menu.jsp 1182
home.jsp 1182
Sample 1 1183
action.jsp 1183
distributor.jsp 1185
full_list.jsp 1186
Screenshot of sample1 view 1186
Sample 2 1187
action.jsp 1187
distributor.jsp 1188
add_form.jsp 1189
full_list.jsp 1190
Screenshot of sample2 view 1191
Using iframe for extending frontend UI 1192
Exposed frontend JavaScript API 1193
General functions 1193
Cryptography functions 1195
Applet functions 1196
HTML Editor functions 1196
Wiki functions 1196
How to expose Java methods to JavaScript 1196
Static method sample 1196
Non Static method sample 1197
Integration sample 1197
Integrate preview with other web applications 1199
Troubleshooting 1202
Samples 1203
Scripting samples 1204
Script - Basic samples 1206
Update repository statistics 1206
Create some folders and set property group 1206
Show number of documents, folders and size from a given path 1206
Get path by UUID 1206
Show all documents used in every running workflow 1207
Script - Compact documents with size equals to 0 1208
Script - Convert TIF to PDF 1209
Script - Count all children subfolder 1210

Script - Exporter 1211
Script - Export repository folder to server as a ZIP file 1212
Script - Folders deep 1213
Images 1214
Execution 1214
Results 1214
Script - Import data from CSV file 1216
Image 1218
Register the metadata group 1218
Repository view 1218
Execution 1219
Script results 1220
Imported metadata 1220
Script - Purge all folders into some parent 1221
Script - Purge all users trashes 1222
Script - Put documents from folder and descendants into pending text extraction queue 1223
Script - Read PDF form fields 1224
Script - Recursive character renaming 1225
Script - Recursive repository traversal 1226
Script - Recursive change security 1227
Script - Security change at low level 1229
Images 1232
Initial security status 1232
UUID of the folder 1233
Execution 1234
Log 1235
Rebuild Lucene indexes 1236
Security changed applied 1237
Script samples for automation 1239
Script for automation - Execute OMR 1240
Images 1240
Register automation rule 1240
Templates defintion 1241
Result 1242
Crontab samples 1244
Crontab sample - Basic file system document importer 1245
Register a new plugin 1246
Images 1246
Crontab task 1246
Files in the application server 1247
Result 1247
Crontab sample - Document importer with metadata values 1249
Register a new plugin 1254
Images 1254
Register metadata definition 1254
Crontab task 1255
Files in application server 1256
Result 1256
Mail notification 1256
Imported files into the repository 1257
Crontab task - XML importer 1259
Register a new plugin 1264
Images 1264
Crontab task 1264
Register metadata definition 1265
Logfile folder 1266

Output folder 1266
Result 1266
Variables by automation event 1268
Available environment variables 1268
Document 1271
Create Document 1271
Update Document 1272
Delete Document 1273
Rename Document 1273
Purge Document 1273
Move Document 1274
Restore Version Document 1274
Download Document from UI 1275
Download Document from UI for preview 1276
Print Document 1276
Stamp Document 1277
Virus detected Document 1277
Set Node Class Document 1278
Folder 1278
Create Folder 1278
Delete Folder 1279
Move Folder 1280
Export Folder as ZIP 1280
Export Folder as JAR 1281
Mail 1281
Create mail 1281
Move Mail 1282
Download mail from UI 1283
Set Node Class Mail 1283
Mail import attachments Mail 1284
Record 1284
Create record 1284
Delete record 1285
Move Record 1285
Set Node Class Record 1286
Property Group (metadata) 1286
Add property group 1286
Set property group 1287
Remove property group 1288
Auth 1288
Create user 1288
User login 1289
User logout 1289
Task 1289
Create Task 1289
Update Task 1290
Delete Task 1290
Note 1290
Create Note 1291
Update Note 1291
Delete Note 1292
General 1292
Text extraction 1292
Export nodes to zip 1293
Vault of stuff 1294
Convert Exchange Legacy DN 1295
Download document with direct link 1297
Import document with bash script using REST 1298
Creating users from scripts 1299
API description 1300
OKMActivity 1302
Methods 1302

findByActivity 1302
countByFilter 1302
getActions 1303
OKMAuth 1305
Basics 1305
Methods 1305
login 1305
login 1306
grantUser 1307
revokeUser 1308
getGrantedUsers 1308
grantRole 1309
revokeRole 1310
getGrantedRoles 1310
getUsers 1311
getRoles 1312
getUsersByRole 1312
getRolesByUser 1313
changeSecurity 1313
createUser 1314
deleteUser 1315
updateUser 1315
createRole 1316
deleteRole 1316
updateRole 1317
assignRole 1317
removeRole 1318
getProfiles 1318
getUserProfile 1319
setUserProfile 1320
logout 1320
getUser 1321
setUserPermissions 1321
setRolePermissions 1322
getUserTenants 1323
setUserTenant 1323
OKMBookmark 1325
Basics 1325
Methods 1325
add 1325
get 1326
remove 1326
rename 1327
getAll 1327
OKMDashboard 1329
Basics 1329
Methods 1329
getUserCheckedOutDocuments 1329
getUserLastModifiedDocuments 1329
getUserLockedDocuments 1330
getUserLockedRecords 1331
getUserSubscribedDocuments 1331
getUserSubscribedFolders 1332
getUserSubscribedRecords 1332
getUserLastCreatedDocuments 1333
getUserLastCreatedFolders 1334
getUserLastCreatedRecords 1334
getUserLastDownloadedDocuments 1335
getUserLastImportedMails 1335
getUserLastImportedMailAttachments 1336
getUserDocumentsSize 1336
getUserSearchs 1337
find 1338
getLastWeekTopDownloadedDocuments 1339
getLastMonthTopDownloadedDocuments 1339
getLastWeekTopModifiedDocuments 1340

getLastMonthTopModifiedDocuments 1340
getLastModifiedDocuments 1341
getLastCreatedDocuments 1342
getLastCheckedOutDocuments 1342
getLastCreatedFolders 1343
getLastCreatedRecords 1343
visiteNode 1344
OKMDocument 1346
Basics 1346
Methods 1346
create 1346
createSimple 1347
delete 1348
getProperties 1348
getContent 1349
getContentByVersion 1350
getExtractedText 1351
getChildren 1351
rename 1352
setProperties 1353
setLanguage 1353
setTitle 1354
checkout 1355
cancelCheckout 1355
forceCancelCheckout 1356
isCheckedOut 1356
checkin 1357
checkin 1358
lock 1359
unlock 1359
forceUnlock 1360
isLocked 1361
getLockInfo 1361
purge 1362
move 1362
copy 1363
copy 1364
extendedCopy 1364
getVersionHistorySize 1366
isValid 1366
getPath 1367
getDetectedLanguages 1367
createFromTemplate 1368
updateFromTemplate 1369
setAnnotations 1370
getAnnotations 1371
getCheckedOut 1371
stamp 1372
setDescription 1373
setNodeClass 1373
setDispositionStage 1374
createWizard 1374
getNumberOfPages 1375
getPageAsImage 1376
mergePdf 1376
OKMFolder 1378
Basics 1378
Methods 1378
create 1378
createSimple 1379
getProperties 1379
delete 1380
purge 1380
rename 1381
move 1382
copy 1382
copy 1383

extendedCopy 1384
extendedCopy 1385
getChildren 1386
getContentInfo 1386
isValid 1387
getPath 1388
setStyle 1388
createMissingFolders 1389
addNodeClassChildren 1389
setDescription 1390
OKMImport 1391
Basics 1391
Methods 1391
createDocument 1391
createFolder 1392
importFolder 1392
OKMLink 1394
Methods 1394
create 1394
getChildren 1394
getLinks 1395
isValid 1396
OKMMail 1397
Basics 1397
Methods 1397
getProperties 1397
createAttachment 1398
deleteAttachment 1398
getAttachments 1399
delete 1399
purge 1400
rename 1401
move 1401
copy 1402
copy 1402
extendedCopy 1403
extendedCopy 1404
getChildren 1405
isValid 1406
getPath 1406
sendMail 1407
setTitle 1407
sendMailWithAttachments 1408
importEml 1409
importMsg 1410
setDocumentNodeClass 1411
setDispositionStage 1411
setDescription 1412
getContent 1412
createWizard 1413
forwardEmail 1414
OKMNode 1416
Basics 1416
Methods 1416
getNodeByUuid 1416
getVersionHistory 1417
restoreVersion 1417
purgeVersionHistory 1418
mayBePromotedAsRecord 1418
promoteAsRecord 1419
degradeRecord 1420
isElectronicRecordPath 1420
getElectronicRecordInPath 1421
getChildrenPaginated 1421
getChildrenByCategoryPaginated 1423

getBreadcrumb 1424
subscribe 1425
unsubscribe 1425
importZip 1426
unzip 1426
exportZip 1427
getNodesFromUuids 1428
evaluateDownloadZip 1428
restore 1429
hasNodesLockedByOtherUser 1430
lock 1431
forceLock 1431
unlock 1432
forceUnlock 1432
isLocked 1433
getLockInfo 1434
setComment 1434
OKMNote 1436
Basics 1436
Methods 1436
addNote 1436
get 1437
delete 1437
set 1438
list 1439
getNoteHistories 1439
OKMNotification 1441
Basics 1441
Methods 1441
subscribe 1441
subscribe 1442
getSubscriptors 1442
notify 1443
OKMProperty 1444
Basics 1444
Methods 1444
addCategory 1444
removeCategory 1445
addKeyword 1445
removeKeyword 1446
setEncryption 1447
unsetEncryption 1447
setSigned 1448
OKMPropertyGroup 1450
Basics 1450
Methods 1450
addGroup 1450
removeGroup 1452
getGroups 1452
getAllGroups 1453
getProperties 1454
setProperties 1454
getPropertyGroupForm 1456
hasGroup 1456
getSuggestions 1457
registerDefinition 1458
getRegisteredDefinition 1459
getGroups 1460
getGroupProperties 1460
getProperties 1461
OKMRecord 1462
Basics 1462
Methods 1462
create 1462

createSimple 1463
getProperties 1463
delete 1464
purge 1464
rename 1465
move 1466
copy 1466
copy 1467
extendedCopy 1468
isValid 1469
getPath 1469
getChildren 1470
lock 1470
unlock 1471
forceUnlock 1472
setTitle 1472
setNodeClass 1473
setDispositionStage 1473
addNodeClassChildren 1474
setDescription 1474
createWizardRecord 1475
OKMRelation 1477
Basics 1477
Methods 1477
getRelationTypes 1477
add 1478
delete 1479
getRelations 1479
getRelationGroups 1480
addGroup 1480
addNodeToGroup 1481
deleteGroup 1482
getGroup 1482
setGroupName 1483
OKMReport 1485
Basics 1485
Methods 1485
list 1485
get 1486
execute 1486
OKMRepository 1488
Methods 1488
getRootFolder 1488
getTrashFolder 1488
getTrashFolderBase 1489
getTemplatesFolder 1490
getPersonalFolder 1490
getPersonalFolderBase 1491
getMailFolder 1491
getMailFolderBase 1492
getThesaurusFolder 1493
getCategoriesFolder 1493
purgeTrash 1494
getUpdateMessage 1494
getRepositoryUuid 1495
hasNode 1495
getNodePath 1496
getNodeUuid 1497
getAppVersion 1497
copyAttributes 1498
getMetadataFolder 1499
getChangeLog 1499
getAppLicense 1500
OKMSearch 1501
Basics 1501

Methods 1504
find 1504
find 1505
findPaginated 1506
findPaginated 1507
findByQuery 1508
findByQuery 1509
findByQueryPaginated 1509
findByQueryPaginated 1510
saveSearch 1512
updateSearch 1512
getSearch 1513
getAllSearchs 1514
deleteSearch 1514
getKeywordMap 1515
getCategorizedDocuments 1516
getCategorizedFolders 1516
getCategorizedMails 1517
getCategorizedRecords 1518
getDocumentsByKeyword 1518
getFoldersByKeyword 1519
getMailsByKeyword 1519
getRecordsByKeyword 1520
getDocumentsByPropertyValue 1520
getFoldersByPropertyValue 1521
getMailsByPropertyValue 1521
getRecordsByPropertyValue 1522
findMailPaginated 1523
findPaginated 1524
findPaginated 1526
findWithMetadata 1527
findWithMetadataPaginated 1528
findWithMetadataPaginated 1529
findByQueryPaginated 1531
findByQueryPaginated 1532
OKMStamp 1534
Methods 1534
calculateStampCoordinates 1534
calculateStampExpressions 1535
stampDocument 1536
stampImageCustom 1536
stampImageCustom 1537
getAllStamps 1538
getStampTextByPk 1539
getStampImageByPk 1539
calculateFlyImageDimensions 1540
getPersonalStamps 1540
getPersonalStampImage 1541
OKMStats 1543
Basics 1543
Methods 1543
getDocumentsByContext 1543
getDocumentsSizeByContext 1543
getFoldersByContext 1544
getMailsByContext 1544
getRecordsByContext 1545
OKMTask 1547
Basics 1547
Methods 1549
create 1549
update 1550
find 1551
delete 1551
getByProject 1552
getByType 1552
getAssigned 1553

getAssigned 1554
countAssigned 1555
getActive 1556
getActive 1557
countActive 1558
getFinished 1559
getFinished 1560
countFinished 1561
findStatus 1562
findProject 1563
findType 1563
findNote 1564
getStatuses 1564
getProjects 1565
getTypes 1565
getNotes 1566
createStatus 1566
createProject 1567
createType 1568
createNote 1568
updateStatus 1569
updateProject 1570
updateType 1570
updateNote 1571
deleteStatus 1572
deleteProject 1572
deleteType 1573
deleteNote 1573
getByNode 1574
getAssigned 1574
countAssigned 1576
getActive 1577
countActive 1579
getFinished 1580
countFinished 1581
getAssignedCount 1582
getActiveCount 1583
getFinishedCount 1584
OKMUserConfig 1586
Methods 1586
setHome 1586
getConfig 1586
OKMWorkflow 1588
Methods 1588
registerProcessDefinition 1588
deleteProcessDefinition 1588
getProcessDefinitionImage 1589
getProcessDefinitionForms 1590
runProcessDefinition 1590
sendProcessInstanceSignal 1591
endProcessInstance 1592
deleteProcessInstance 1593
findProcessInstances 1593
findAllProcessDefinitions 1594
findLatestProcessDefinitions 1594
findLastProcessDefinition 1595
findAllProcessDefinitionVersions 1596
getProcessInstance 1596
suspendProcessInstance 1597
resumeProcessInstance 1597
addProcessInstanceVariable 1598
deleteProcessInstanceVariable 1598
findUserTaskInstances 1599
findPooledTaskInstances 1600
findTaskInstances 1601
setTaskInstanceValues 1601
addTaskInstanceComment 1602

getTaskInstance 1603
setTaskInstanceActorId 1603
addTaskInstanceVariable 1604
deleteTaskInstanceVariable 1604
startTaskInstance 1605
endTaskInstance 1605
suspendTaskInstance 1606
resumeTaskInstance 1607
getToken 1607
addTokenComment 1608
suspendToken 1608
resumeToken 1609
sendTokenSignal 1609
setTokenNode 1610
endToken 1611
Commons classes 1612
ArchiveUtils 1614
Methods 1614
createZip 1614
createJar 1614
ExecutionUtils 1616
Methods 1616
runScript 1616
runJar 1616
runCmd 1617
FileLogger 1618
Methods 1618
info 1618
warn 1618
error 1619
FileUtils 1620
Methods 1620
getFileName 1620
getFileExtension 1620
FormatUtil 1622
Methods 1622
formatSize 1622
parseSize 1622
formatDate 1623
ISO8601 1624
Methods 1624
parseBasic 1624
formatBasic 1624
GeneralUtils 1626
Methods 1626
uiNotify 1626
MailUtils 1627
Methods 1627
sendMessage 1627
sendMessage 1627
sendDocuments 1628
PathUtils 1630
Methods 1630
getParent 1630
getName 1630
getContext 1631
PDFUtils 1632
Methods 1632
merge 1632
stampImage 1633
stampText 1634

getNumberOfPages 1635
PrincipalUtils 1636
Methods 1636
getUser 1636
getTenantId 1636
StackTraceUtils 1638
Methods 1638
isCallingMe 1638
toString 1638
TemplateUtils 1640
Methods 1640
templateExists 1640
replace 1640
WorkflowUtils 1642
Methods 1642
findProcessInstancesByNode 1642
ShardUtils 1643
Methods 1643
setCurrentShard 1643
createShardDirs 1643
changeNodeShard 1644
getShardIds 1644
getShardSize 1645
getShardPaths 1645
getShards 1646
getShardFolders 1646
LegacySrv 1648
executeSQL 1648
executeHQL 1648
Database description 1650
Node types 1650
SQL example queries 1651
Mark documents to being indexed again 1651
Node version and properties 1651
SQL example queries 1653
Node categories and keywords 1654
SQL example queries 1655
Advanced samples 1656
Custom UI template 1657
General description 1657
Properties file 1657
Authentication 1658
Log file 1658
Remember me 1658
Roles management 1658
Bootstrap 1659
Packaging 1659
Application port 1659
Catalog information 1659
Description 1659
Installation 1659
Step 1: Download 1660
Step 2: Role creation 1660
Step 3: SQL data needed 1660
Step 4: Metadata 1660
Step 5: Automations 1661
Form UI template 1663
Packaging 1663

Installation 1663
Step 1 - register metadata into OpenKM 1663
Step 2 - upload template file 1663
template.html 1665
Security Provider 1666
General description 1666
Properties file 1666
Custom Principal Adapter 1666
Step 1: Download 1666
Step 2: OpenKM configuration 1667
Step 3: Install the plugin 1667
Authentication Provider 1667
Step 1: Download 1667
Step 2: Install the patch 1667
Step 3: OpenKM configuration 1668
Set unique document name based on metadata 1669

OpenKM 7.1

OpenKM Professional version is licensed under terms of "**EULA - OpenKM End User License Agreement**".

For more information take a look at Licenses.

OpenKM is an Electronic Document and Record Management System (EDRMS). There's a lot of literature about document management terms like: DMS, EDRMS or CMS usually more influenced by marketing rules rather than objective reasons.

The differences between DMS, CMS, and EDRMS in most cases are subjective or not relevant from the Document Management point of view. Lately, it seems the term CMS has gotten some marketing success and actually is one of the most common used. However, we prefer the term **EDRMS**, perhaps with lesser success on the marketing world but from our point of view more strict and representative.

OpenKM helps you to:

- Day-to-day data capture.
- Storage.
- Modify and share physical and/or digital documents within the organization.
- Apply retention periods to stored items.
- Identify the owner of each record series and documents.
- Set a chain of custody and proper audit trail.
- When needed apply legal records holds.
- Managing records disposition.
- Preserving records through their life cycle.

Some of the OpenKM goals are:

- Increase **efficiency**. Faster documents approval, reduced manual data entry, automated recurring tasks among others.
- Set **compliance** levels. The system helps the organization to avoid penalties when regulators, auditors and other government authorities summons them for inspections.
- Preserve **historical content**. Follow the status of the document (determined by the different phases of the lifecycle).
- **Consolidate** the company knowledge on a single access point.

The OpenKM technical staff maximum is "**Try finding out the easy solution for each problem. If you cannot find it, try again.**"

Although we've tried to give our powerful application what it needs to solve problems with a minimalist solution, sometimes its not possible. OpenKM can solve day-to-day complex scenarios. It is not a closed application and has a lot of features that can be used and configured on several ways.

Please take your time to read the documentation, it is time well spent. If you have some doubt contact with us.

Usually a problem have several ways to be resolved.

Your suggestions are always welcomed, we try creating a better application to solve real life problems in a more efficient way. Your suggestions help us growing the knowledge of EDRMS solutions and how it can be applied to real case scenarios.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Hardware and software requirements

This section presents information about the environment needed to run an OpenKM server, listing the required and optional software and providing some facts about their configuration.

This sections aims at helping you to validate or define the production environment. However, the check-list is not exhaustive and needs to be completed with the user's experience.

Requirements

OpenKM can be installed on any operating system (Linux, Windows, Mac OS X,) since it is a multi-platform application.

A Linux host is a good choice because this OS handles heavy I/O applications, it works with OpenKM quite well, and it is conveniently priced.

The only requirement to run the OpenKM is to have **Java 8** (also called Java 1.8) installed.

Make sure you're installing OpenJDK 1.8 if you install JRE the application will not run correctly.

Checking the Java version

1. Open a terminal:
    - On Linux or Mac OS X: **open a terminal**.
    - On Windows: press "windows" key + r, type **cmd** (or command) in the Run window and press "OK" or open the "Prompt command" from "Start > Programs > Accessories" menu.
2. Type **java -version** and press **Enter**.

   If Java is correctly installed on your computer, the name and version of the Java virtual machine is displayed:

```
Java version "1.8.0_31"
Java(TM) SE Runtime Environment (build 1.8.0_31-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.31-b07, mixed mode)
```

If Java is not installed on your computer, it will fail to display the Java version. Then, you will need to install Java (see below).

If Java is installed but not included in the PATH, it will fail to find the Java command. Then, you will need to add $JAVA_HOME/bin/ in your PATH (see: How do I set or change the PATH system variable?).

Installing Java

For Windows users

If the required version of Java is not installed on your computer:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

1. Download it from the Oracle website (Java 8) and choose the appropriate platform for your hardware and Windows version.
2. Run the downloaded .exe file and follow the instructions displayed.

For Linux users

You can download Java 8 from Oracle website (choose the right architecture). You can find RPM and TGZ packages to install.

If you use Ubuntu the following is a better method:

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
```

This way your OpenJDK will be upgraded every time a new release is available.

In case you have another OpenJDK installed (like GCJ), you can select the right one this way:

```
$ sudo update-alternatives --config java
```

For Mac OS X Users

Java packages and instructions for installation are available from the Oracle website:

- Download
- Instructions

Recommendations

Hardware configuration

OpenKM runs quite well on any computer with a common hardware configuration. For a better performance we recommend at least 4 GB RAM (have at least >=2GB of plenty RAM) and a fast hard disk like SATA or better.

A dual-core Intel based CPU with 3.20 GHz should be fine for a reduced number of users (30-50 concurrent users).

Be sure to have enough disk space for the document repository and its versions.

A Linux host is a good choice because this OS handles heavy I/O applications like this quite well, and it is not expensive.

It is highly recommended 64 bits architecture.

We suggest having a modern server with good performance.

Although OpenKM can be installed on a server with other applications, the best scenario is a dedicated server. Keep in mind that on a shared scenario other applications can have a negative effect on OpenKM.

For more information about hardware requirements consult with your sales & marketing representatives as they should provide you a hardware requirements estimation.

Try giving them as much detailed information as possible for a more accurate estimation:

- Estimated number of concurrent users.
- Type of users (% of reader and % of writers).
- Estimated first year number of files.
- Estimated number of files growing per year.
- Estimated document size average.
- Number of document in the initial import
- Are these documents uploaded by users o by an automatic process?
- Most common document types (MS Office, OpenOffice, PDF, etc.).

You can use your older history of documents and folders to get some of these values.

Browser support

OpenKM user interface is based on GWT framework, than means the supported browsers are determinated by GWT browser support.

Actual browsers supported:

- Firefox
- Internet Explorer 8, 9, 10, 11
- Safari 5, 6
- Chromium and Goggle Chrome
- Opera latest version

More information at GWT Browsers and Servers.

We encourage your browser to be updated to the latest version.

We recommend Firefox and Chrome rather than IE, for several reasons; best performance, less security holes, follow the standards.

Operating systems

Linux (Known working configurations):

- Ubuntu

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- Debian
- CentOS
- Red Hat Linux

Red Hat Linux is a more difficult scenario, because some packages aren't by default in the Red Hat official repositories.

Ubuntu is the preferred Linux choise. We run the tests against Linux distribution and we build the packages for it. However, any recent Linux distribution should be usable as long as:

- OpenJDK 8 is available.
- You can adapt the provided Linux scripts.
- Thirdparty software requirements

Windows (Known working configurations):

- For development: Windows 7, Windows 8
- For deployment: Windows Server 2003, Windows Server 2008, Windows Server 2012

Databases

Known working configurations:

- MySQL
- PostgreSQL
- Oracle
- SQL server

The application internally uses Hibernate, in fact you should get it running from any of the supported Hibernate dialects, for more information read Hibernate: SQL Dialects.

JVM

Although OpenKM can run with other JDK we recommend the use of OpenJDK 8 (64 bits architecture is highly recommended).

LDAP

- Open LDAP
- Microsoft Active Directory

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Third-party software requirements

OpenKM takes advantage of using third-party software for some features. Some of this software needs to be installed on the server, others are optional.

Software	Usage	Additional information	Required
Openoffice or Libreoffice	• Converts documents to pdf. • Needed for preview feature.		Yes
Imagemagick	• Transforms images. • Generates thumbnails. • Needed for preview feature.	Usually comes by default installed in Linux OS. In Windows in most cases it is not necessary to install it, because it comes embedded by default with OpenKM bundle installation.	Yes
SWFTools	• Converts pdf document to swf. • Needed for preview feature.	On most cases it is not necessary to install it, because it comes embedded by default with OpenKM bundle installation.	Yes
GhostScript	• Interpreter for the PostScript language and for PDF. • Needed by Imagemagick.	Usually comes by default installed on Linux OS.	Optional
Pdftohtml	• Converts pdf document to html. • Needed for preview feature.		Optional

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Tesseract or cuneiform	• OCR engine. • Used for the text extraction feature.	Required when application must processes images to extract text (OCR).	Optional
Ffmpeg	• Converts and streams audio and video		Optional
Clamav or equivalent	• Antivirus	Required when antivirus feature is enabled.	Optional

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Installation

OpenKM can be installed on several Operating Systems and can be configured to use different databases. Because it's implemented in Java, the only software requirement is having a Java Virtual Machine compiled for your platform. But OpenKM also depends on third-party applications to be fully configured. For example, it needs a working OpenOffice or LibreOffice installation to be able of preview documents. Or you may want to extract text from your scanned images, and for this an OCR engine is needed.

Before installing you need to prepare the server. Depending on the Operating System you should follow some tips. When you install the Operating System, keep on mind that OpenKM does not need a graphical user desktop. So, in Linux systems, you can avoid installing these packages and save some memory. Another important advice is to do a minimal Operating System installation. We only need the minimum to access the server. In case of Linux server, we prefer to access by SSH because is fast and secure. Please, don't install Java or Tomcat or any other software unless you have already asked us.

This is a checklist of information we should know before installing:

- How many document do you want to store?
- How many user will access OpenKM?
- How many concurrent users?
- How much memory does the server have?
- How many CPUs does the server have?
- How much free disk is avaible in the server?
- Which Operating System is installed?
- Is the installed Operating System the 64 bits version?

In case of Linux server there are some aditional questions:

- Which distro (name and version)?
- Does the server have a GUI (X-Windows) installed?
- It is a clean and minimal installation (only SSH) or have you installed another software?
- Is this server used for other applications?
- Is the database already installed?

In this section you will learn how to install OpenKM in your server and how to match most common third-party applications requirements.

OpenKM has been successfully installed on:

OS	Description

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux	Ubuntu Server Linux Mint Debian CentOS Red Hat SuSE
Windows	Windows Server 2012 R2 Windows Server 2008 R2 Windows Server 2003 R2 Windows 8 Windows 7

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Installing on Ubuntu and Debian

Preliminaries

In case to configure OpenKM in a cluster or sharding mode, please before starting with the installation process read the sections:

- Cluster configuration
- Sharding configuration

Check disk size

```
$ df -h
```

Create user named openkm

```
$ sudo adduser openkm
```

You can easily generate random passwords at https://www.random.org/passwords.

Check if partners channels are enabled

```
$ cat /etc/apt/sources.list
```

In case they are not enabled, enable them.

```
$ sudo vim /etc/apt/sources.list
$ sudo apt-get update
```

Check server configuration

```
$ wget -Nc smxi.org/inxi
$ sudo chmod +x inxi
$ sudo ./inxi -F
```

Increase ulimit

```
$ sudo vim /etc/pam.d/su
```

uncomment the following line

```
#session    required   pam_limits.so
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

to

```
session    required   pam_limits.so
```

```
$ sudo vim /etc/security/limits.conf
```

and add the following lines to the end of the file (before the line # End of file)

```
*   soft  nofile   6084
*   hard  nofile   6084
```

Now reboot the server, execute **ulimit -n** and check if the open files limits are now updated.

Checking Java version

```
$ java -version
```

If Java is correctly installed on your computer, the name and version of the Java virtual machine is displayed:

```
openjdk version "1.8.0_212"
OpenJDK Runtime Environment (build 1.8.0_212-8u212-b03-0ubuntu1.18.04.1-b03)
OpenJDK 64-Bit Server VM (build 25.212-b03, mixed mode)
```

The Java version may be different depending on your installed one, but we always recommend the latest 1.8 release.

Install Java

Perform these steps only if Java 8 is not already installed on your server.

In case of Ubuntu

```
$ sudo apt-get update
$ sudo apt-get install openjdk-8-jdk
```

In case of Debian

```
$ su -
$ apt-get update
$ apt-get install openjdk-8-jdk
$ exit
```

In case of Debian 10 "Buster", Java 8 packages are not included in the default repositories so you need to switch to SID (which

is not recommendable for production servers) or use the packages from https://adoptopenjdk.net/. Run these sentences as root:

```
$ apt install wget gnupg software-properties-common
$ wget -qO - https://adoptopenjdk.jfrog.io/adoptopenjdk/api/gpg/key/public | apt-key add -
$ add-apt-repository --yes https://adoptopenjdk.jfrog.io/adoptopenjdk/deb/
$ apt-get update
$ apt-get install adoptopenjdk-8-hotspot
```

Set OpenJDK 1.8 as your default Java version

Perform this only if Java 8 is not your default Java version.

```
$ sudo update-alternatives --config java
```

Installing database

Install MySQL if you do not have it already installed:

```
$ sudo apt-get install mysql-server
```

When you install MySQL, the installer will ask you for a password for the root user. Keep this password handy.

Also you can install MariaDB, which replaces MySQL in latest versions:

```
$ sudo apt-get install mariadb-server
```

Change MySQL root password

Do it only if it's necessary to change MySQL root password.

Method 1

```
$ /usr/bin/mysqladmin -u root -h localhost password 'password'
```

Method 2

```
$ sudo /etc/init.d/mysql stop
$ sudo mysqld --skip-grant-tables &
$ mysql -u root mysql
> UPDATE user SET Password=PASSWORD('YOURNEWPASSWORD') WHERE User='root';
> FLUSH PRIVILEGES;
> exit;
```

MySQL in this case has not been started at service. The process must be killed to stop MySQL and start it again as a service.

Method 3

```
$ mysql
> UPDATE mysql.user SET Password=PASSWORD('MyNewPass') WHERE User='root';
> flush privileges;
```

Method 4

```
$ sudo /etc/init.d/mysql stop
$ mysqld_safe --skip-grant-tables &
$ mysql -u root mysql
> UPDATE user SET Password=PASSWORD('YOURNEWPASSWORD') WHERE User='root';
> FLUSH PRIVILEGES;
> exit;
```

MySQL in this case has not been started at service. The process must be killed to stop MySQL and start it again as a service

More information at MySQL : Reseting permissions.

Check InnoDB is default MySQL engine

```
$ mysql -h localhost -u root -p
> show engines;
```

It should looks something like this:

| **InnoDB** | **DEFAULT** | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |

If it not default, modify **/etc/mysql/my.cnf**

```
$ vim /etc/mysql/my.cnf
```

and under [mysqld] add

```
default-storage-engine = innodb
```

After changing default MySQL engine the MySQL service must be restarted to take effect.

Create database

```
CREATE DATABASE okmdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;

CREATE USER openkm@localhost IDENTIFIED BY 'password';

GRANT ALL ON okmdb.* TO openkm@localhost WITH GRANT OPTION;
```

Installing Tomcat

```
$ cd /home/openkm

$ unzip Tomcat-8.5.24.zip
```

The tomcat version numeration name can be altered in order to upgrade to a major version.

Configuring Tomcat as a service

For security reasons you shouldn't run Tomcat as **root**. It is better to use the user named openkm.

Create a file with the script:

```
$ sudo vim /etc/init.d/tomcat
```

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:          tomcat
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start and stop Apache Tomcat
# Description:       Enable Apache Tomcat service provided by daemon.
### END INIT INFO

ECHO=/bin/echo
TEST=/usr/bin/test
TOMCAT_USER=openkm
TOMCAT_HOME=/home/openkm/tomcat-8.5.24
TOMCAT_START_SCRIPT=$TOMCAT_HOME/bin/startup.sh
TOMCAT_STOP_SCRIPT=$TOMCAT_HOME/bin/shutdown.sh

$TEST -x $TOMCAT_START_SCRIPT || exit 0
$TEST -x $TOMCAT_STOP_SCRIPT || exit 0

start() {
    $ECHO -n "Starting Tomcat"
    su - $TOMCAT_USER -c "$TOMCAT_START_SCRIPT &"
    $ECHO "."
}

stop() {
    $ECHO -n "Stopping Tomcat"
    su - $TOMCAT_USER -c "$TOMCAT_STOP_SCRIPT 60 -force &"
    while [ "$(ps -fu $TOMCAT_USER | grep java | grep tomcat | wc -l)" -gt "0" ]; do
        sleep 5; $ECHO -n "."
```

```
    done
    $ECHO "."
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        stop
        sleep 30
        start
        ;;
    *)
        $ECHO "Usage: tomcat {start|stop|restart}"
        exit 1
esac
exit 0
```

The script must be changed if you use other user to start the service or your $TOMCAT_HOME is not /home/openkm/tomcat-8.5.24

Make it executable:

```
$ sudo chmod 755 /etc/init.d/tomcat
```

Update the run-levels:

```
$ sudo update-rc.d tomcat defaults
```

Check the service

Start the service:

```
$ sudo service tomcat start
```

Stop the service:

```
$ sudo service tomcat stop
```

Installing OpenKM

With tomcat stopped copy the OpenKM.war file into /home/openkm/tomcat-8.5.24/webapps

```
$ cp OpenKM.war /home/openkm/tomcat-8.5.24/webapps
```

Installing third-party software

Software	Required	Command

LibreOffice or OpenOffice	Yes	$ apt-get install libreoffice
Tesseract	No	$ apt-get install tesseract-ocr tesseract-ocr-eng Depending on your locale you would like to install other language file.
Clamav	No	$ apt-get install clamav
Imagemagick	Yes	$ apt-get install imagemagick
Ghostsctipt	Yes	$ apt-get install ghostscript

Start application

Check openkm.properties parameters

```
$ vim /home/openkm/tomcat-8.5.24/openkm.properties
```

Your configuration should be something like this:

```
# OpenKM Hibernate configuration values
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect
spring.jpa.hibernate.ddl-auto=create-only
```

Ensure **spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect** is enabled if your database is MySQL.

First time application start:

```
$ service tomcat start
```

To see the startup process can execute the command:

```
$ tail -f /home/openkm/tomcat-8.5.24/log/catalina.log
```

When application is started you'll see in log file the lines:

```
2015-07-04 18:28:10,680 [main] INFO  org.apache.coyote.http11.Http11Protocol - Starti
2015-07-04 18:28:10,688 [main] INFO  org.apache.coyote.ajp.AjpProtocol - Starting Pro
2015-07-04 18:28:10,692 [main] INFO  org.apache.catalina.startup.Catalina - Server st
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The application will be accessible at http://YOUR_IP:8080/OpenKM by user:**okmAdmin** with password:**admin**. (Do not forget to change it !). For more information about Manage users and roles.

Configure default extensions:

Go to Administration > Utilities > Database query and execute:

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('808e7a42-2e73-470c-ba23-e4c9d
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('58392af6-2131-413b-b188-1851a
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '808e7a42-2e
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '58392af6-21
```

Other common additional configurations setup

OpenKM comes with a lot of configuration parameters. Please make sure to review: Recommended configuration parameters.

- Configuring Apache HTTP Reverse-Proxy
- Configuring mail
- Configuring Tomcat memory utilization
- Configuring Tomcat port
- Configuring Tomcat SSL/TLS
- Configuring Tomcat timezone

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Installing on RedHat and CentOS

If you decide installing Redhat, take in consideration you should have a licensed Redhat version, otherwise the repositories for installing software are locked.

Does not have much sense installing Redhat without a license subscription in the same direction does not have much sense enable CentOS repos in a Redhat, if this is your way better intalling CentOS at the begining.

Preliminaries

In case to configure OpenKM in a cluster or sharding mode, please before starting with the installation process read the sections:

- Cluster configuration
- Sharding configuration

Check disk size

```
$ df -h
```

Create a user named openkm

```
$ sudo adduser openkm
```

You can easily generate random passwords at https://www.random.org/passwords.

Check server configuration

```
$ wget -Nc smxi.org/inxi
$ sudo chmod +x inxi
$ sudo ./inxi -F
```

If it doesn't work, try installing this package:

```
$ sudo yum install pciutils
```

Increase ulimit

```
$ sudo vim /etc/security/limits.conf
```

add the following lines to the end of the file (before the line # End of file)

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
*    soft   nofile    6084
*    hard   nofile    6084
```

Execute **ulimit -n** and see the open files limits are now updated.

Checking Java version

```
$ java -version
```

If Java is correctly installed on your computer, the name and version of the Java virtual machine is displayed:

```
openjdk version "1.8.0_212"
OpenJDK Runtime Environment (build 1.8.0_212-b04)
OpenJDK 64-Bit Server VM (build 25.212-b04, mixed mode)
```

The Java version numeration name can be altered in order to be upgraded to major version.

Install Java

Perform these steps only if Java 8 is not already installed on your server.

```
$ sudo yum install java-1.8.0-openjdk-devel
```

The Java version may be different depending on your installed one, but we always recommend the lastest 1.8 release.

Set OpenJDK 1.8 as your default Java version

Perform this only if Java 8 is not your default Java version.

```
$ alternatives --config java
```

Installing database

Install MySQL if you do not have it already installed:

```
$ sudo yum install mysql-server mysql
```

Configure it as service:

```
$ sudo chkconfig --levels 235 mysqld on
```

On latest CentOS the MySQL package has been replaced in the repositories by MariaDB.

Install MariaDB:

```
$ sudo yum install mariadb-server
```

Configure as service:

```
$ sudo systemctl enable mariadb
```

Start database:

```
$ sudo systemctl start mariadb
```

If you get some trouble on the first database startup because it can not create files into tmp folder do:

Edit the /etc/selinux/config file:

```
$ sudo vim /etc/selinux/config
```

And apply this change to:

```
'SELINUX=disabled'
```

To take effect inmediatelly, execute from command line:

```
$ sudo setenforce 0
```

Change MySQL root password

Do it only If it's necessary to change the MySQL root password.

Method 1

```
$ /usr/bin/mysqladmin -u root -h localhost password 'password'
```

Method 2

```
$ sudo /etc/init.d/mysql stop
$ sudo mysqld --skip-grant-tables &
$ mysql -u root mysql
> UPDATE user SET Password=PASSWORD('YOURNEWPASSWORD') WHERE User='root';
> FLUSH PRIVILEGES;
> exit;
```

In this case MySQL has not been started as a service. The process must be killed to stop MySQL and start it again as service.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method 3

```
$ mysql
> UPDATE mysql.user SET Password=PASSWORD('MyNewPass') WHERE User='root';
> flush privileges;
```

Method 4

```
$ sudo /etc/init.d/mysql stop
$ mysqld_safe --skip-grant-tables &
$ mysql -u root mysql
> UPDATE user SET Password=PASSWORD('YOURNEWPASSWORD') WHERE User='root';
> FLUSH PRIVILEGES;
> exit;
```

In this case MySQL has not been started as service. The process must be killed to stop MySQL and start it again as service.

More information at MySQL : Reseting permissions.

Check InnoDB is default MySQL engine

```
$ mysql -h localhost -u root -p
> show engines;
```

It should show something like this:

| **InnoDB** | **DEFAULT** | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |

If it not default, modify **/etc/mysql/my.cnf**

```
$ vim /etc/mysql/my.cnf
```

and under [mysqld] add

```
default-storage-engine = innodb
```

After changing the default MySQL engine the MySQL service must be restarted to take effect.

Create database

```
CREATE DATABASE okmdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;
CREATE USER openkm@localhost IDENTIFIED BY 'password';
```

```
GRANT ALL ON okmdb.* TO openkm@localhost WITH GRANT OPTION;
```

Installing Tomcat

```
$ cd /home/openkm

$ unzip Tomcat-7.0.61.zip
```

The tomcat version numeration name can be altered in order to be upgraded to major version.

Configuring Tomcat as a service

For security reasons you shouldn't run Tomcat as **root**. It is better to use the user named openkm.

Create a file with the script:

```
$ sudo vim /etc/init.d/tomcat
```

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:          tomcat
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start and stop Apache Tomcat
# Description:       Enable Apache Tomcat service provided by daemon.
### END INIT INFO

ECHO=/bin/echo
TEST=/usr/bin/test
TOMCAT_USER=openkm
TOMCAT_HOME=/home/openkm/tomcat-7.0.61
TOMCAT_START_SCRIPT=$TOMCAT_HOME/bin/startup.sh
TOMCAT_STOP_SCRIPT=$TOMCAT_HOME/bin/shutdown.sh

$TEST -x $TOMCAT_START_SCRIPT || exit 0
$TEST -x $TOMCAT_STOP_SCRIPT || exit 0

start() {
    $ECHO -n "Starting Tomcat"
    su - $TOMCAT_USER -c "$TOMCAT_START_SCRIPT &"
    $ECHO "."
}

stop() {
    $ECHO -n "Stopping Tomcat"
    su - $TOMCAT_USER -c "$TOMCAT_STOP_SCRIPT 60 -force &"
    while [ "$(ps -fu $TOMCAT_USER | grep java | grep tomcat | wc -l)" -gt "0" ]; do
        sleep 5; $ECHO -n "."
    done
    $ECHO "."
}

case "$1" in
```

```
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        stop
        sleep 30
        start
        ;;
    *)
        $ECHO "Usage: tomcat {start|stop|restart}"
        exit 1
esac
exit 0
```

The script must be changed if you use other user to start the service or your $TOMCAT_HOME is not /home/openkm/tomcat-7.0.61

Make it executable:

```
$ sudo chmod 755 /etc/init.d/tomcat
```

Update the run-levels:

```
$ sudo chkconfig tomcat --level 2345 on
```

Check the service

Start the service:

```
$ sudo service tomcat start
```

Stop the service:

```
$ sudo service tomcat stop
```

Installing OpenKM

With tomcat stopped copy the OpenKM.war file into /home/openkm/tomcat-7.0.61/webapps

```
$ cp OpenKM.war /home/openkm/tomcat-7.0.61/webapps
```

Installing thirdparty software

Software	Required	Description
OpenOffice or LibreOffice	Yes	Check these packages are present: • openoffice.org-**ure**-3.1.1-19.5.el5_5.6

		• openoffice.org-**headless**-3.1.1-19.5.el5_5.6 • openoffice.org-**pyuno**-3.1.1-19.5.el5_5.6 Execute this command line to check the installed packages: `$ rpm -qa | grep openoffice` Take the names of the packages as an orientation. If headless package is not installed OpenKM will not be able to start soffice service.
Tesseract	No	Some useful links: • https://code.google.com/p/tesseract-ocr/wiki/Compiling • https://code.google.com/p/python-tesseract/wiki/HowToCompileForCentos • http://www.vicchiam.com/blog/?p=168
ClamAV	No	Some useful links: • http://www.clamav.net/doc/install.html#rhel
Imagemagick	Yes	`$ yum install ImageMagick`
GhostScript	Yes	`$ yum install ghostscript`
Htop	no	Download from http://pkgs.repoforge.org/htop/ `$ rpm -ivH htop-1.0.3-1.el6.rf.x86_64.rpm`

Start application

Check openkm.properties parameters

```
$ vim /home/openkm/tomcat-7.0.61/openkm.properties
```

Your configuration should be something like this:

```
# OpenKM Hibernate configuration values
```

```
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto=create-only
```

Ensure **spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect** is enabled if your database is MySQL

First time application start:

```
$ service tomcat start
```

To check if the startup process can execute the command:

```
$ tail -f /home/openkm/tomcat-7.0.61/log/catalina.log
```

When the application is started you'll see in log file the lines:

```
2015-07-04 18:28:10,680 [main] INFO  org.apache.coyote.http11.Http11Protocol - Starti
2015-07-04 18:28:10,688 [main] INFO  org.apache.coyote.ajp.AjpProtocol - Starting Pro
2015-07-04 18:28:10,692 [main] INFO  org.apache.catalina.startup.Catalina - Server st
```

The application will be accessible at http://YOUR_IP:8080/OpenKM by user:**okmAdmin** with password:**admin**. (Do not forget to change it !). For more information please check: Manage users and roles.

Configure the default extensions:

Go to Administration > Utilities > Database query and execute:

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('808e7a42-2e73-470c-ba23-e4c9d
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('58392af6-2131-413b-b188-1851a
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '808e7a42-2e
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '58392af6-21
```

Linux Oracle 7.X Throubleshooting

For "**OpenKM as service**" and "**Imagemagick**" issues check this section RedHat and CentOS issues

The libreoffice and tesseract tools are not into the default repositories, you cal follow the steps described in the URL below for installing them:

Install libreoffice from RPM

You should download the last RPM file version, what might be found at https://www.libreoffice.org/download/download/?type=rpm-x86&version=6.0.4&lang=en

The process description below has been described at https://www.tecmint.com/install-libreoffice-on-rhel-centos-fedora-debian-ubuntu-linux-mint/

Libreoffice at the end will be installed into the folder /opt/libreoffice6.0

```
$ yum remove openoffice* libreoffice*

$ cd /opt

$ wget https://www.libreoffice.org/donate/dl/rpm-x86_64/6.0.4/es/LibreOffice_6.0.4_Linux_x86-64_rpm.tar.gz

$ tar -xvf LibreOffice_6.0.4_Linux_x86-64_rpm.tar.gz

$ cd LibreOffice_6.0.4_Linux_x86-64_rpm/RPMS

$ yum localinstall *.rpm
```

Install Tesseract

The script below has been found from https://github.com/EisenVault/install-tesseract-redhat-centos/ we suggest take a look at the last version of the script in the github project for updates.

Before executing the script, review it, consider to apply changes about downloading the dictionaries you need, for example for spanish dictionary should be download https://github.com/tesseract-ocr/tessdata/raw/3.04.00/spa.traineddata what is not present in the script by default.

```
#!/bin/sh

cd /opt

yum -y update
yum -y install libstdc++ autoconf automake libtool autoconf-archive pkg-config gcc gc

#Install AutoConf-Archive
wget ftp://mirror.switch.ch/pool/4/mirror/epel/7/ppc64/a/autoconf-archive-2016.09.16-
rpm -i autoconf-archive-2016.09.16-1.el7.noarch.rpm

#Install Leptonica from Source
wget http://www.leptonica.com/source/leptonica-1.75.3.tar.gz
tar -zxvf leptonica-1.75.3.tar.gz
cd leptonica-1.75.3
./autobuild
./configure
make
make install
cd ..

#!/bin/sh

#Install Tesseract from Source
wget https://github.com/tesseract-ocr/tesseract/archive/3.05.01.tar.gz
tar -zxvf 3.05.01.tar.gz
cd tesseract-3.05.01
./autogen.sh
PKG_CONFIG_PATH=/usr/local/lib/pkgconfig LIBLEPT_HEADERSDIR=/usr/local/include ./conf
LDFLAGS="-L/usr/local/lib" CFLAGS="-I/usr/local/include" make
make install
ldconfig
cd ..

#Download and install tesseract language files
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/ben.traineddata
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/eng.traineddata
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.traineddata
```

```
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/tha.traineddata
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/osd.traineddata
mv *.traineddata /usr/local/share/tessdata

#Download Hindi Cube data
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.bigrams
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.fold
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.lm
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.nn
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.params
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.cube.word-freq
wget https://github.com/tesseract-ocr/tessdata/raw/3.04.00/hin.tesseract_cube.nn
mv hin.* /usr/local/share/tessdata

ln -s /opt/tesseract-3.05.01 /opt/tesseract-latest
```

Other common additional setup configurations

OpenKM comes with a lot of configuration parameters we suggest taking a look at: Recommended configuration parameters.

- Configuring Apache HTTP Reverse-Proxy
- Configuring mail
- Configuring Tomcat memory utilization
- Configuring Tomcat port
- Configuring Tomcat SSL/TLS
- Configuring Tomcat timezone

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Installing on Windows

Preliminaries

In case to configure OpenKM in a cluster or sharding mode, please before starting with the installation process read the sections:

- Cluster configuration
- Sharding configuration

Check that you have enought disk size.

We recommend installing these tools:

- Geany (text editor).
- mTail (Windows Program that works like the Unix command "tail").

Checking Java version

```
C:\> java -version
```

If Java is correctly installed on your computer, the name and version of the Java virtual machine is displayed:

```
java version "1.8.0_101"
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)
```

The Java version may be different depending on your installed one, but we always recommend the latest 1.8 release.

We recommend also setting the JAVA_HOME variable:

```
setx JAVA_HOME "C:\Program Files\Java\jdk1.8.0_101" /M
```

Install Java

Do these steps only if java 8 is not installed on your server.

If the required version of Java is not installed on your computer:

1. Download it from the Oracle website (Java 8) and choose the appropriate platform for your hardware and Windows version.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

2. Run the downloaded .exe file and follow the instructions displayed.

Installing database

Install MySQL if you do not have it already installed:

Download MySQL MSI Installer from MySQL Community Server and follow the installation wizard steps.

If you do not have a MySQL client software it is also good idea to install MySQL Workbench.

Check InnoDB is default MySQL engine

```
show engines;
```

It should show something like this:

| **InnoDB** | **DEFAULT** | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |

If it not default, modify **c:\ProgramData\MySQL\MySQL Server 5.6\my.ini**

and under [mysqld] add

```
default-storage-engine = innodb
```

After changing default MySQL engine the MySQL service must be restarted to take effect.

Create database

```
CREATE DATABASE okmdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;
CREATE USER openkm@localhost IDENTIFIED BY 'password';
GRANT ALL ON okmdb.* TO openkm@localhost WITH GRANT OPTION;
```

Installing Tomcat

Unzip Tomcat-7.0.61.zip to c:\

The Tomcat version numeration name can be altered in order to be upgraded to major version.

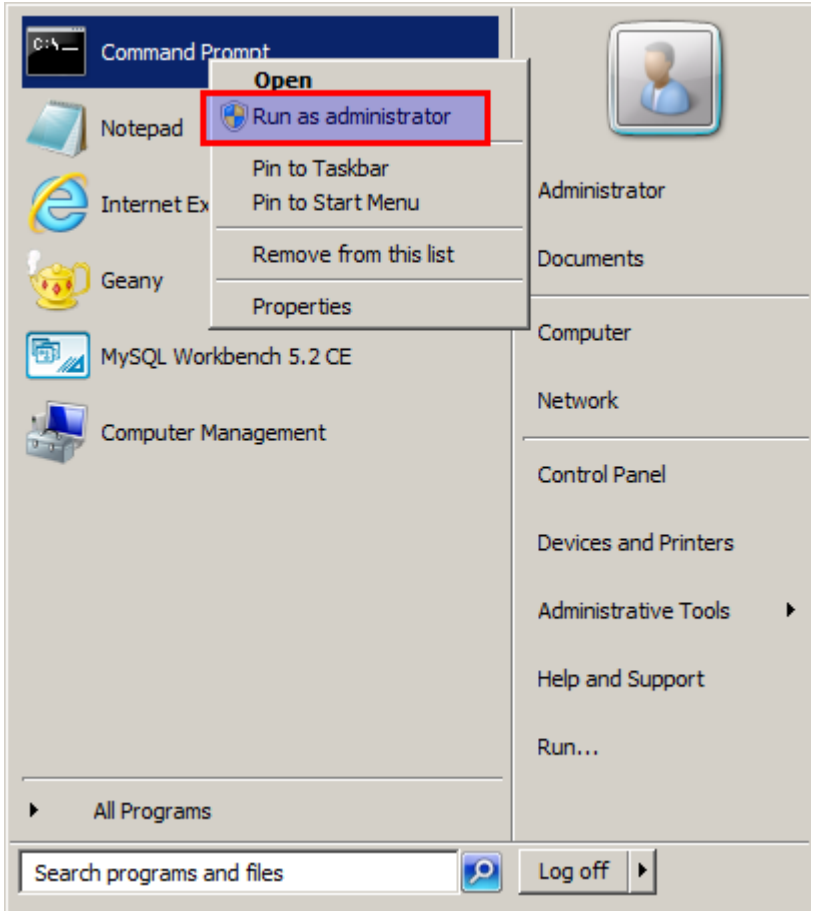
Copy files from **C:\tomcat-7.0.61\bin\x86** or **C:\tomcat-7.0.61\bin\x64** to **C:\tomcat-7.0.61\bin** based on the server architecture.

Configuring Tomcat as a service

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Register the service:

Open a windows terminal as administrator.



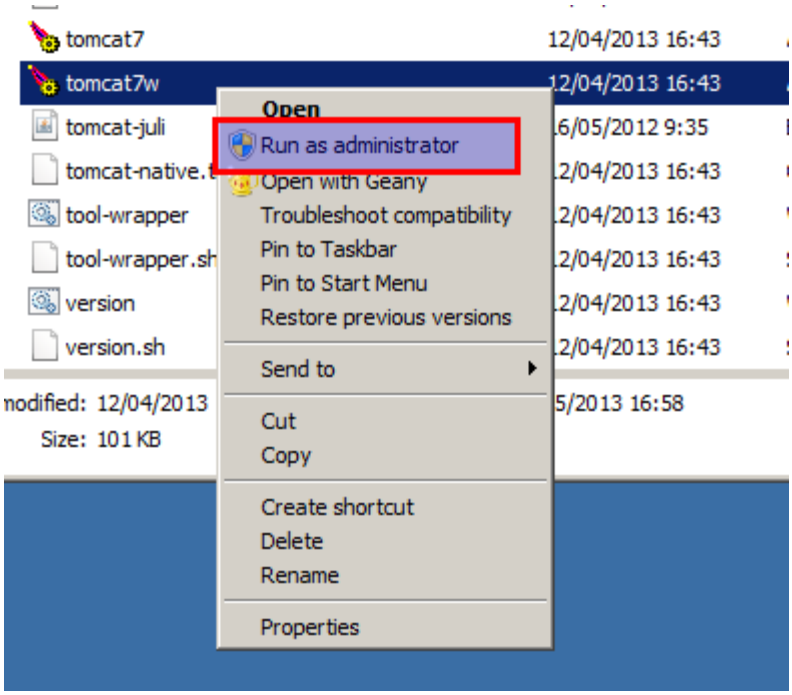

```
cd c:\tomcat-7.0.61\bin
service.bat install
```

Set the JVM parameters for the Tomcat service:

```
cd c:\tomcat-7.0.61\bin
openkm //US//OpenKM ++JvmOptions="-XX:+UseG1GC"
openkm //US//OpenKM ++JvmOptions="-XX:+UseStringDeduplication"
openkm //US//OpenKM ++JvmOptions="-XX:+HeapDumpOnOutOfMemoryError"
openkm //US//OpenKM ++JvmOptions="-Djava.awt.headless=true"
openkm //US//OpenKM ++JvmOptions="-Djava.net.preferIPv4Stack=true"
openkm //US//OpenKM ++JvmOptions="-Dfile.encoding=utf-8"
openkm //US//OpenKM ++JvmOptions="-Dsun.jnu.encoding=UTF-8"
openkm //US//OpenKM ++JvmOptions="-Dmail.mime.ignoreunknownencoding=true"
```

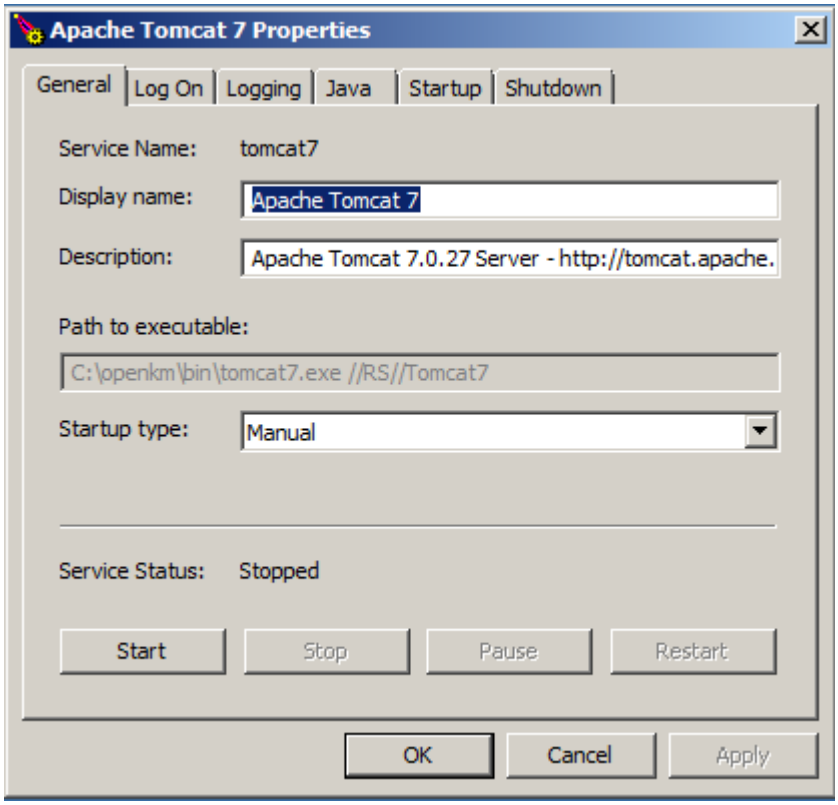
```
openkm //US//OpenKM ++JvmOptions="-Dsun.java2d.cmm=sun.java2d.cmm.kcms.KcmsServiceProvider"
```

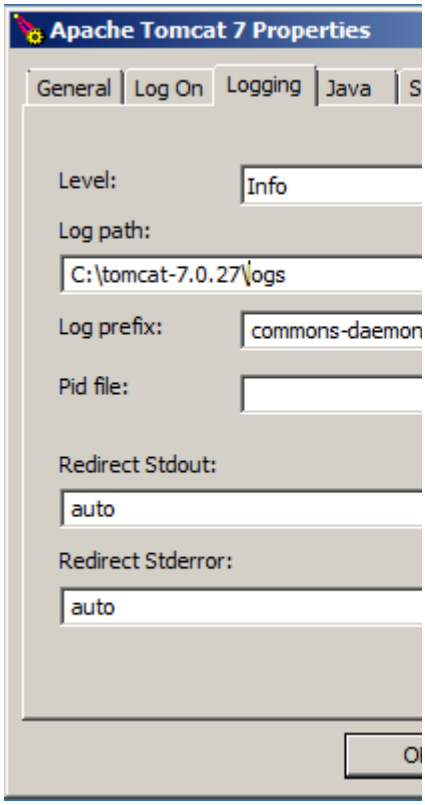
Execute the tomcat7w.exe as administrator:

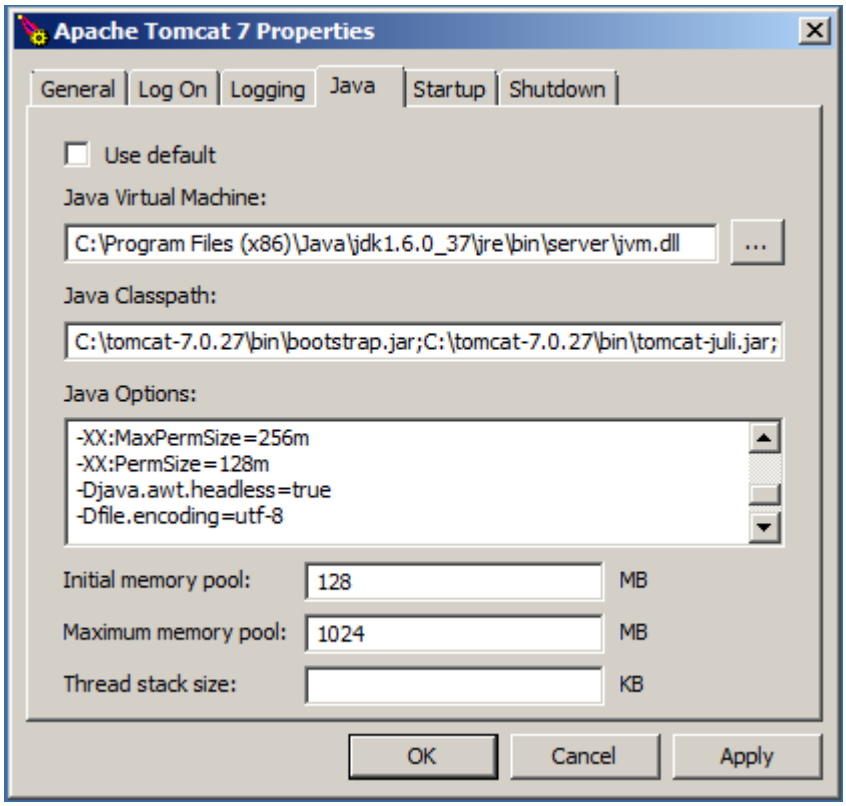


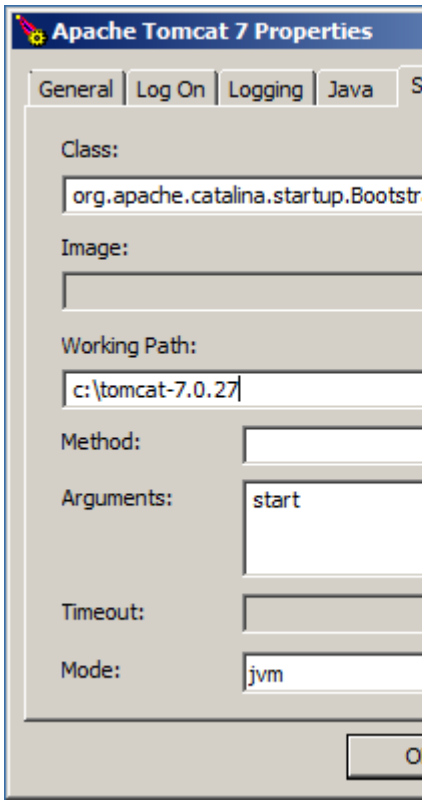
Check the configuration parameters:

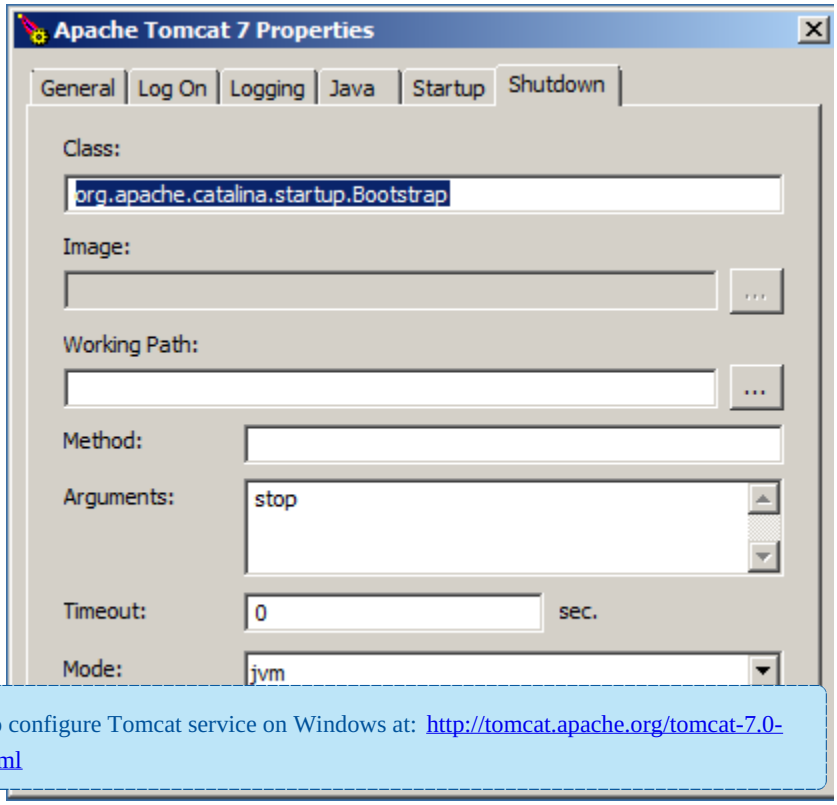
Also you can change "Startup type" to Automatic.


Apache Tomcat 7 Properties
General
Log On
Logging
Java
Startup
Shutdown
Service Name:
tomcat7
Display name:
Apache Tomcat 7
Description:
Apache Tomcat 7.0.27 Server - http://tomcat.apache.
Path to executable:
C:\openkm\bin\tomcat7.exe //RS//Tomcat7
Startup type:
Manual
Service Status:
Stopped
Start
Stop
Pause
Restart
OK
Cancel
Apply


Apache Tomcat 7 Properties
General
Log On
Logging
Java
Level:
Info
Log path:
C:\tomcat-7.0.27\logs
Log prefix:
commons-daemon
Pid file:
Redirect Stdout:
auto
Redirect Stderror:
auto


Apache Tomcat 7 Properties
General
Log On
Logging
Java
Startup
Shutdown
Use default
Java Virtual Machine:
C:\Program Files (x86)\Java\jdk1.6.0_37\jre\bin\server\jvm.dll
Java Classpath:
C:\tomcat-7.0.27\bin\bootstrap.jar;C:\tomcat-7.0.27\bin\tomcat-juli.jar;
Java Options:
-XX:MaxPermSize=256m
-XX:PermSize=128m
-Djava.awt.headless=true
-Dfile.encoding=utf-8
Initial memory pool:
128
MB
Maximum memory pool:
1024
MB
Thread stack size:
KB
OK
Cancel
Apply


Apache Tomcat 7 Properties
General
Log On
Logging
Java
Class:
org.apache.catalina.startup.Bootstr
Image:
Working Path:
c:\tomcat-7.0.27
Method:
Arguments:
start
Timeout:
Mode:
jvm



More information about how to configure Tomcat service on Windows at: http://tomcat.apache.org/tomcat-7.0-doc/windows-service-howto.html

Check the service

From terminal

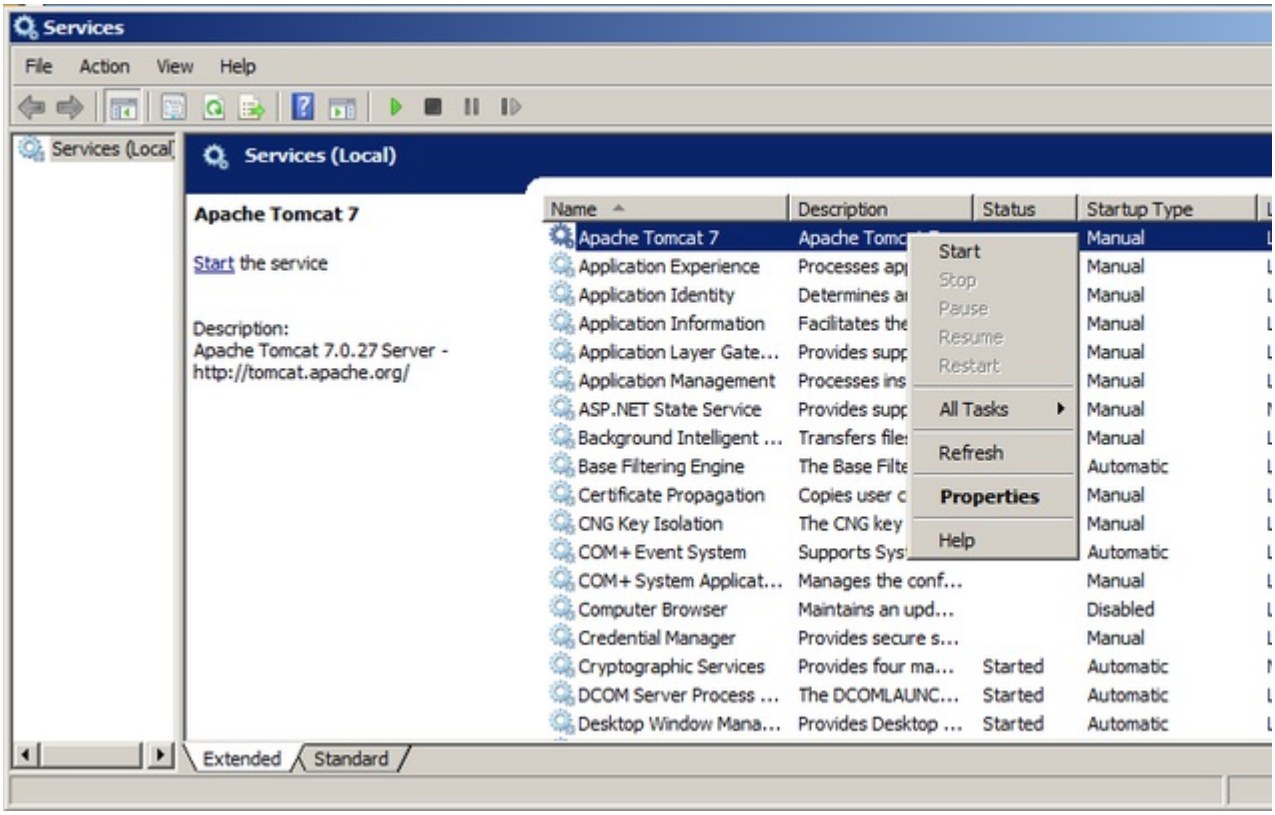
Start the service:

```
$ net start tomcat7
```

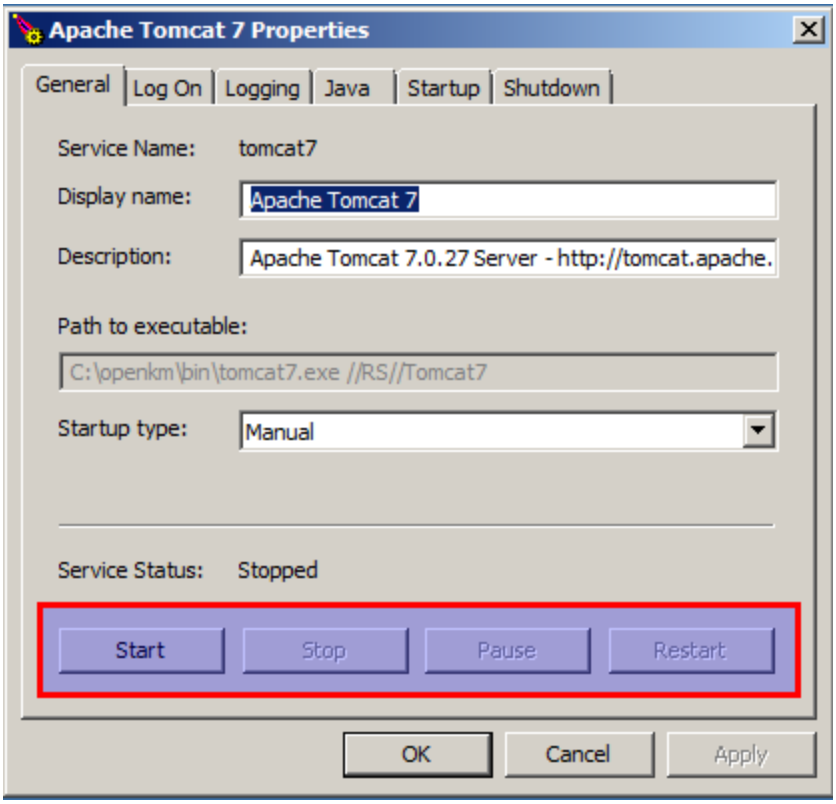
Stop the service:

```
$ net stop tomcat7
```

From Windows service panel



From tomcat7w tool

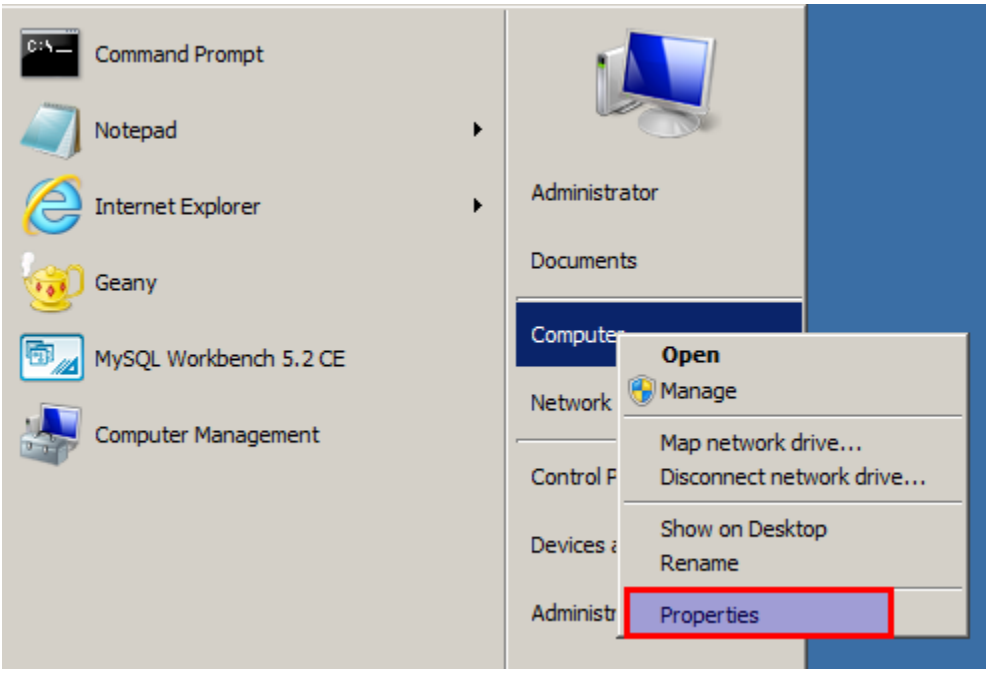


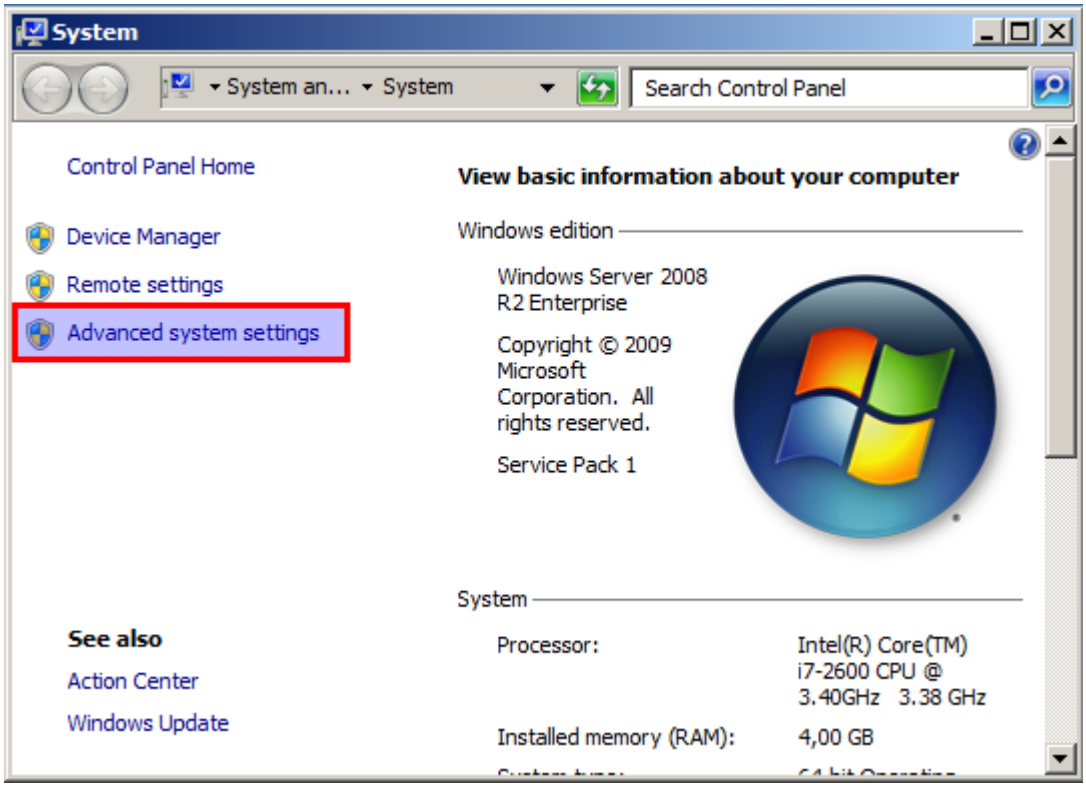
Configuring environment variables

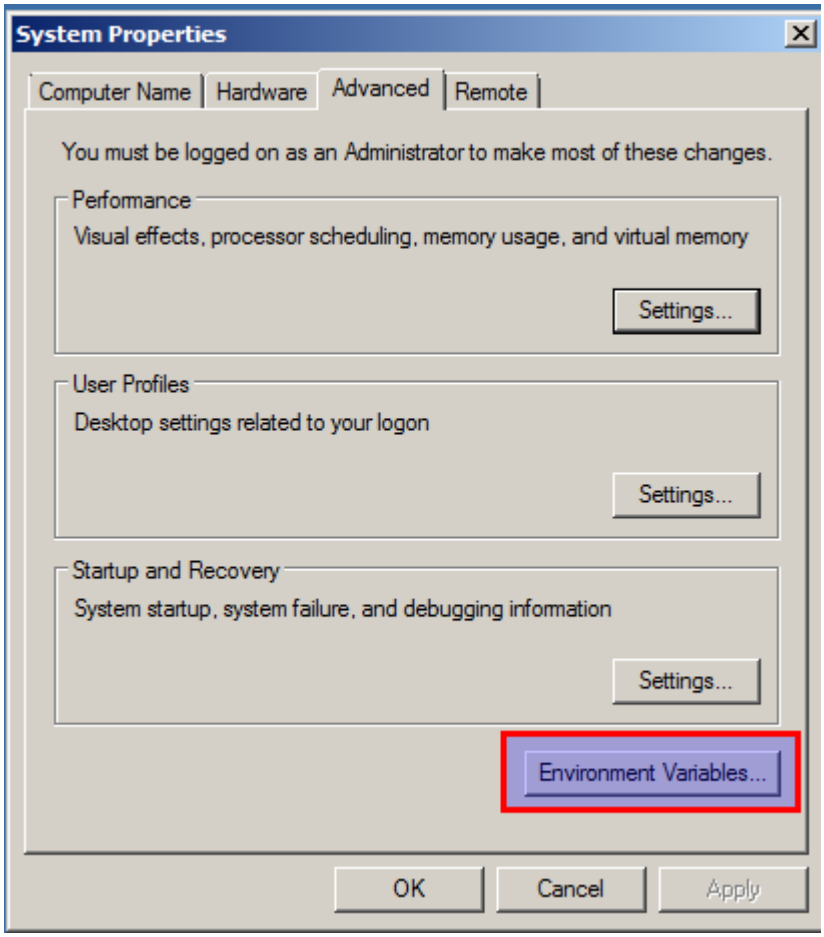
Add to environment path variable the **sigar** folder, the **bin** folder:

```
setx PATH "%PATH%;C:\tomcat-7.0.61\lib\sigar" /M
setx PATH "%PATH%;C:\tomcat-7.0.61\bin" /M
```

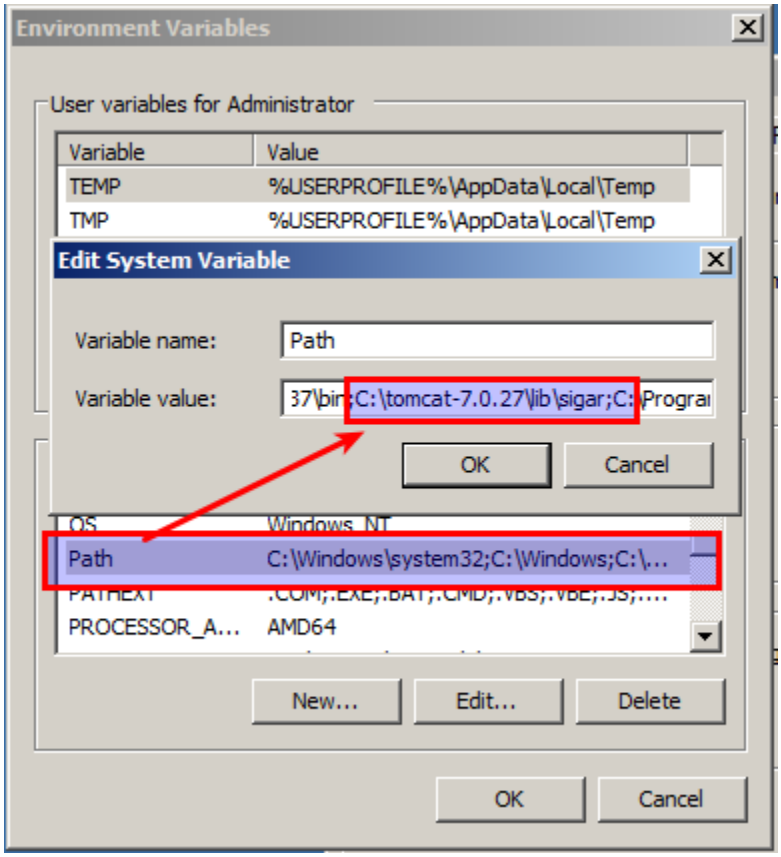
These operations also can be done from Windows environment variables:




System
System an... System
Search Control Panel
Control Panel Home
Device Manager
Remote settings
Advanced system settings
View basic information about your computer
Windows edition
Windows Server 2008 R2 Enterprise
Copyright © 2009 Microsoft Corporation. All rights reserved.
Service Pack 1
System
Processor:
Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz 3.38 GHz
Installed memory (RAM):
4,00 GB
See also
Action Center
Windows Update


System Properties
Computer Name
Hardware
Advanced
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your logon
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply

Add sigar folder path - c:\tomcat-7.0.61\lib\sigar\ - to your environment path variable:



Installing OpenKM

With Tomcat stopped copy the OpenKM.war file into /home/openkm/tomcat-7.0.61/webapps

```
$ copy OpenKM.war c:\tomcat-7.0.61\webapps
```

Install third party software:

Software	Required	Description
OpenOffice	Yes	We recommend using OpenOffice rather than Libreoffice, because we've found problems in some servers to setup as service by OpenKM.
Tesseract	No	Download from https://code.google.com/p/tesseract-ocr/ and follow the installation wizard.
Clamav	No	Download from http://www.clamav.net/download.html and follow the installation wizard.

Imagemagick	No	Tomcat bundle comes with embedded convert tool. In case you decide for install it, go to http://www.imagemagick.org/script/binary-releases.php and download the latest binary release and follow the installation wizard. Take in consideration 16 bits needs more hardware resources than 8 bits. We suggest ImageMagick-6.9.1-7-**Q8**-x64-dll.exe rather than ImageMagick-6.9.1-7-**Q16**-x64-dll.exe (note the version numeration can be distinct due ImageMagick version upgrade).
GhostScript	No	Tomcat bundle comes with embedded Ghostscript tool. If you decide installing it, go to http://ghostscript.com/download/gsdnld.html and follow the installation wizard.

There is also an available **Extras.zip** file that contains all these software as a Portable edition.

Start application

Check openkm.properties parameters

```
C:\> geany c:\openkm\openkm.properties
```

Your configuration should be something like this:

```
# OpenKM Hibernate configuration values
hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
hibernate.hbm2ddl=create-only


# Initial configuration - Windows
system.openoffice.path=C:/Program Files (x86)/OpenOffice 4
```

Ensure **hibernate.dialect=org.hibernate.dialect.MySQL5Dialect** is enabled if your database is MySQL.

First time application start:

```
$ net start tomcat7
```

Use mTail Program - or equivalent - to watch the c:\tomcat-7.0.61\log\catalina.log file

When application is started you'll see it in the log file the lines:

```
2015-07-04 18:28:10,680 [main] INFO  org.apache.coyote.http11.Http11Protocol - Starti
2015-07-04 18:28:10,688 [main] INFO  org.apache.coyote.ajp.AjpProtocol - Starting Pro
2015-07-04 18:28:10,692 [main] INFO  org.apache.catalina.startup.Catalina - Server st
```

The application will be accessible at http://YOUR_IP:8080/OpenKM by user:**okmAdmin** with password:**admin**. (Do not forget to change it !). For more information: Manage users and roles.

Configure default extensions:

Go to Administration > Utilities > Database query and execute:

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('808e7a42-2e73-470c-ba23-e4c9d
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('58392af6-2131-413b-b188-1851a
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '808e7a42-2e
INSERT INTO OKM_PROFILE_MSC_EXTENSION (PEX_ID, PEX_EXTENSION) VALUES (1, '58392af6-21
```

Other common additional setup configurations

OpenKM comes with a lot of configuration parameters please take a look at: Recommended configuration parameters.

- Configuring Apache HTTP Reverse-Proxy
- Configuring Tomcat mail
- Configuring Tomcat memory utilization
- Configuring Tomcat ports
- Configuring Tomcat SSL/TLS
- Configuring Tomcat timezone

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Using the installer

Another way of installing OpenKM is using the installer tool. This is a Java application which ease the installation process and works on Ubuntu, Debian, RedHat, CentOS and Windows platforms. Keep on mind that you need a working Java installation prior to use this tool. We recommend OpenJDK 8. Please check these pages for more information about installing Java in your server; look for **Checking Java version** section:

- Installing on RedHat and CentOS
- Installing on Ubuntu and Debian
- Installing on Windows

The installer download several packages from our servers like Tomcat and OpenKM, so it need to connect to http://download.openkm.com. Once the installation is completed and OpenKM is running it should have access to http://update.openkm.com in order to validate the licence.

Once Java 8 is installed you can download the installer from http://download.openkm.com/okm/OKMInstaller.jar. In Linux you can download it from command line:

```
$ wget http://download.openkm.com/okm/OKMInstaller.jar
```

The installer will make the installation in the folder which it runs in, so before running move the OKMInstaller.jar to the location where OpenKM should be installed. In case of running on Linux it also change the files permissions to an user called "openkm" so, if you haven't create such user, please create it now:

```
$ sudo adduser openkm
```

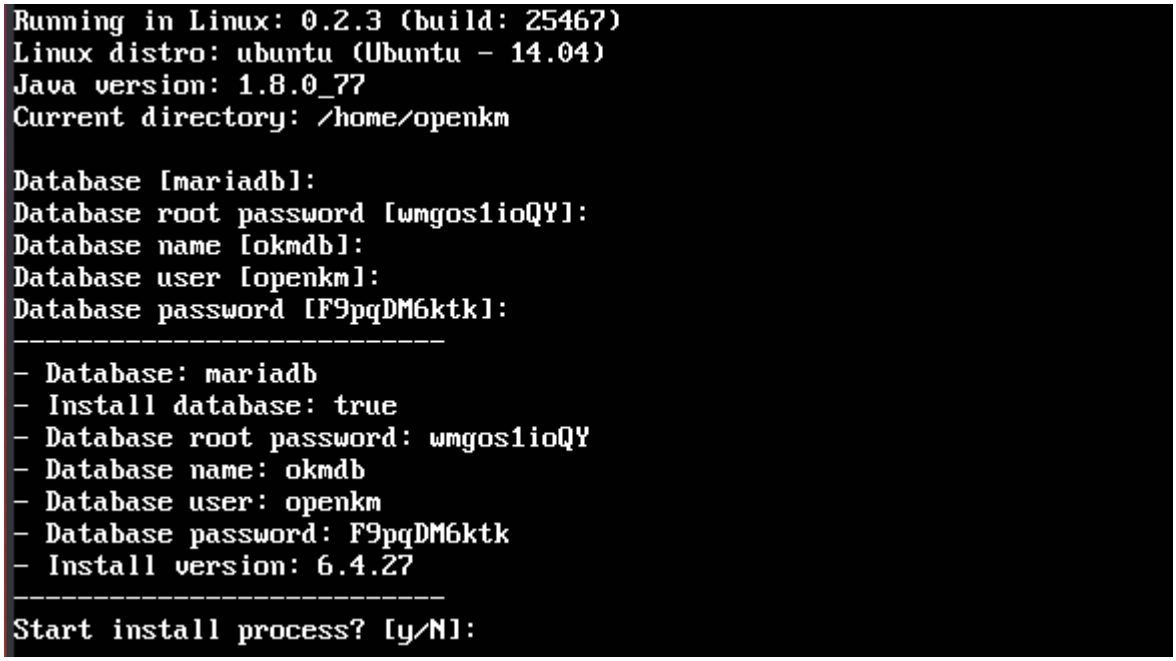
This command should work in most common Linux distributions.

It's a text based installer, so you need to execute it from console as root if you are on Linux or as Administrator if you are on Windows. This is because it needs to install package dependencies and configure OpenKM as a service:

```
$ java -jar OKMInstaller.jar
```

And this is the output:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Values in brackets are the default one. If you hit Enter, this is the chosen value. If you want a different one, just write the string. At the end, the installed will show the entered information and you would be able to verifiy it and proceed or enter again these values.

The installer will ask for an authorized user and password. If you don't have already this information, please open a support ticket at http://support.openkm.com or contact your sales agent.

If you press "y", the installer will ask for an authorized user and password. Once the user and password is validated, the installation process will start. This process may take several minutes depeding on your Internet connection and server performance. In case of Linux based systems, it will install all required dependencies like ImageMagick and LibreOffice.

If you access the Internet using a proxy, just configure it when executing the application:

```
$ java -Dhttp.proxyHost=192.168.0.1 -Dhttp.proxyPort=3128 -jar OKMInstaller.jar
```

You may also need **https.proxyHost** and **https.proxyPort** or even **http.proxyUser**, **http.proxyPassword**, **https.proxyUser** and **https.proxyPassword** parameters.

Supported databases

The installer supports these databases:

- MariaDB (mariadb)
- MySQL (mysql)
- PostgreSQL (posgresql)
- SQL Server (sqlserver)

- Oracle (oracle)
- HSQLDB (hsqldb)
- H2 (h2)

Other options

The installer will also check if the port 8080 is already being used, which is the default port used by Tomcat and should be free.

If you want to disable this check, use the **-p** or **--port** parameters.

```
$ java -jar OKMInstaller.jar -p
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Using Docker

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

There is a Docker image available at our Docker private repository at http://docker.openkm.com, and it is the easiest way yo get OpenKM running on your server. In case you want to use this image, please submit a support ticket and we will provide you access to the repository.

Please, refer to Docker documentation for installation instructions and to learn more about Docker.

Once Docker has been installed, create the proper **openkm.properties** file or configure it using the environment variables and execute this command to get your OpenKM instance:

```
$ docker run --name openkm -p 8080:8080 -v ${PWD}/openkm.properties docker.openkm.com/private/professional
```

Keep on mind you need a database in another container which will be used by the OpenKM container.

And after that, point your browser at http://localhost:8080/. The default user and password is "okmAdmin" / "admin".

This way you can create a container with the default embedded H2 database (not recommended for production, but just fine for testing):

```
$ docker run -d --name openkm -p 8080:8080 -v myvol:/opt/tomcat \
  -e DATABASE_DRIVER=org.h2.Driver \
  -e DATABASE_URL=jdbc:h2:/opt/tomcat/okmdb \
  -e DATABASE_DIALECT=com.openkm.db.dialect.H2Dialect \
  --restart=unless-stopped docker.openkm.com/private/professional:7.1.13
```

If you start a container which creates a new volume, as above, and the container has files in the directory to be mounted (like /opt/tomcat), the directory's contents are copied into the volume. The container then mounts and uses the volume, so we can say the volume initially has a pre-populated content from the image.

This is a sample which can be used to create a container using MySQL, but the recommended way is using Docker Compose:

```
$ docker run -d --name openkm -p 8080:8080 -v myvol:/opt/tomcat \
  -e DATABASE_DRIVER="com.mysql.cj.jdbc.Driver" \
  -e DATABASE_URL="jdbc:mysql://172.17.0.1:3306/dbtest?
useUnicode=true&characterEncoding=UTF8&serverTimezone=CET&nullNamePatternMatchesAll=true" \
  -e DATABASE_USER="openkm" \
```

```
-e DATABASE_PASSWORD="openkm" \
-e DATABASE_DIALECT="com.openkm.db.dialect.MySQL5InnoDBDialect" \
--restart=unless-stopped docker.openkm.com/private/professional:7.1.13
```

The database can be created this way:

```
CREATE USER 'openkm'@'%' IDENTIFIED BY 'openkm';
CREATE DATABASE dbtest DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;
GRANT ALL ON dbtest.* TO 'openkm'@'%' WITH GRANT OPTION;
```

To see the container logs:

```
$ docker logs -f openkm
```

To stop the container:

```
$ docker stop openkm
```

To start it again:

```
$ docker start openkm
```

To access to the container:

```
$ docker exec -ti openkm bash
```

For more actions, please check Docker documentation site.

Configuration

These are the environment variables which can be used to configure the OpenKM container:

Variable	Description
DATABASE_DRIVER	The JDBC database driver to connect to the database.
DATABASE_URL	The JDBC URL to connect to the database.
DATABASE_USER	The database user.
DATABASE_PASSWORD	The database user password.
DATABASE_DIALECT	The Hibernate database dialect.

Docker Compose

To make all this configuration easy, we recommend using Docker Compose. Please, refer to Docker Compose documentation for installation instructions and more info.

This is a sample Docker Compose configuration file using environment variables, where MySQL is used as database:

```
version: '3.2'

services:

  openkm_mysql:
    image: docker.openkm.com/private/professional:7.1.13
    container_name: openkm_mysql
    hostname: openkm
    ports:
      - 8080:8080
    volumes:
      - openkm_mysql:/opt/tomcat
    environment:
      - DATABASE_DRIVER=com.mysql.cj.jdbc.Driver
      - DATABASE_URL=jdbc:mysql://mysql:3306/okmdb?useUnicode=true&characterEncoding=
      - DATABASE_USER=openkm
      - DATABASE_PASSWORD=openkm
      - DATABASE_DIALECT=com.openkm.db.dialect.MySQL5InnoDBDialect
      - TZ=Europe/Madrid
    entrypoint: [ "wait-for-it.sh", "mysql:3306", "--timeout=0", "--strict", "--", "e
    links:
      - mysql:mysql
    networks:
      skynet:
        ipv4_address: 172.28.1.1

  mysql:
    image: mysql:8.0.18
    container_name: okmdb_mysql
    hostname: mysql
    command: --default-authentication-plugin=mysql_native_password --character-set-se
    volumes:
      - okmdb_mysql:/var/lib/mysql
    environment:
      - MYSQL_DATABASE=okmdb
      - MYSQL_USER=openkm
      - MYSQL_PASSWORD=openkm
      - MYSQL_ROOT_PASSWORD=openkm
    security_opt:
      - seccomp:unconfined
    networks:
      skynet:
        ipv4_address: 172.28.1.2

volumes:
  openkm_mysql:
  okmdb_mysql:

networks:
  skynet:
    ipam:
      driver: default
      config:
```

```
        - subnet: 172.28.0.0/16
```

This is another sample Docker Compose configuration:

```
version: '3.2'

services:

  openkm_mysql:
    image: docker.openkm.com/private/professional:7.1.13
    container_name: openkm_mysql
    hostname: openkm
    ports:
      - 8080:8080
    volumes:
      - ${PWD}/openkm.properties:/opt/tomcat/openkm.properties
      - ${PWD}/repository:/opt/tomcat/repository
    environment:
      - TZ=Europe/Madrid
    entrypoint: [ "wait-for-it.sh", "mysql:3306", "--timeout=0", "--strict", "--", "e
    links:
      - mysql:mysql
    networks:
      skynet:
        ipv4_address: 172.28.1.1

  mysql:
    image: mysql:8.0.18
    container_name: okmdb_mysql
    hostname: mysql
    command: --default-authentication-plugin=mysql_native_password --character-set-se
    volumes:
      - ${PWD}/mysql:/var/lib/mysql
    environment:
      - MYSQL_DATABASE=okmdb
      - MYSQL_USER=openkm
      - MYSQL_PASSWORD=openkm
      - MYSQL_ROOT_PASSWORD=openkm
    security_opt:
      - seccomp:unconfined
    networks:
      skynet:
        ipv4_address: 172.28.1.2

networks:
  skynet:
    ipam:
      driver: default
      config:
        - subnet: 172.28.0.0/16
```

This is the **openkm.properties** file:

```
# OpenKM Hibernate configuration values
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://mysql:3306/okmdb?useUnicode=true&characterEncoding=
spring.datasource.username=openkm
spring.datasource.password=openkm

# JPA stuff
spring.jpa.hibernate.ddl-auto=create-only
```

```
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect

# Cluster
cluster.enabled=false
cluster.node=master

# Logging config
spring.output.ansi.enabled=always
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Installing on a pristine Tomcat

In this section, it will be depicted how deploy OpenKM in a pristine Apache Tomcat.

Preliminaries

Take a look to the installation of the whole system according with your Operating System:

- Installing on RedHat and CentOS
- Installing on Ubuntu and Debian
- Installing on Windows

Installing Tomcat

Download latest Tomcat version from Apache Tomcat page (Tomcat 8) and extract it into the directory you want to install it.

At this moment, Tomcat 8.5 is the preferred option because Tomcat 9 support it still in beta.

Once Tomcat has been extracted, some modifications are neccessary.

Copy libraries and configuration files

Copy the following required libraries to TOMCAT_HOME/lib. These files can be found in the Tomcat already configured by the OpenKM team, which can be downloaded from http://download.openkm.com/okm/Tomcat-7.0.61.zip

- unicode.ttf

Copy the following files to TOMCAT_HOME:

- openkm.properties
- logback.xml

Modify configuration files

Modify the following files from TOMCAT_HOME/conf:

setenv.sh or **setenv.bat**

Depending on your Operating System you may have to create a file called **setenv.sh** in Linux or **setenv.bat** in Windows. This file should be placed at **TOMCAT_HOME/bin**. This is the context in case of Linux:

```
JAVA_OPTS="$JAVA_OPTS -Xms512m -Xmx2048m -XX:+UseG1GC -XX:+UseStringDeduplication"
JAVA_OPTS="$JAVA_OPTS -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=$CATALINA_HOME
JAVA_OPTS="$JAVA_OPTS -Djava.net.preferIPv4Stack=true -Dmail.mime.ignoreunknownencodi
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Djava.awt.headl
JAVA_OPTS="$JAVA_OPTS -Dsun.java2d.cmm=sun.java2d.cmm.kcms.KcmsServiceProvider"
CATALINA_PID=$CATALINA_HOME/catalina.pid
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/bin/cadviewer
```

And this is the content in case of Windows:

```
set JAVA_OPTS=%JAVA_OPTS% -Xms512m -Xmx2048m -XX:+UseG1GC -XX:+UseStringDeduplication
set JAVA_OPTS=%JAVA_OPTS% -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=%CATALINA_
set JAVA_OPTS=%JAVA_OPTS% -Djava.net.preferIPv4Stack=true -Dmail.mime.ignoreunknownen
set JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Djava.awt.h
set PATH=%PATH%;%CATALINA_HOME%\bin\cadviewer
```

Installing OpenKM

With Tomcat stopped copy the OpenKM.war file into TOMCAT_HOME \webapps

```
$ mv openkm.war $TOMCAT_HOME/webapps
```

Install third party software

To do this, go to the section corresponding to your Operating System:

- Installing on RedHat and CentOS
- Installing on Ubuntu and Debian
- Installing on Windows

Start application

Check openkm.properties parameters

```
$ vim TOMCAT_HOME/openkm.properties
```

Your configuration file should contain something like this:

```
# OpenKM Hibernate configuration values
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/okmdb?useUnicode=true&characterEnco
spring.datasource.username=openkm
spring.datasource.password=openkm

# JPA stuff
spring.jpa.hibernate.ddl-auto=create
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL57InnoDBDialect

# Logback configuration file
logback.config=logback.xml
```

This configuration works if your database is MySQL. In other case, please check the documentation of create a support ticket.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

First time application start:

```
$ start openkm
```

Use mTail Program - or equivalent - to watch the TOMCAT_HOME\log\catalina.log file

When the application starts, you should see these log messages:

```
2015-07-04 18:28:10,680 [main] INFO  org.apache.coyote.http11.Http11Protocol - Starti
2015-07-04 18:28:10,688 [main] INFO  org.apache.coyote.ajp.AjpProtocol - Starting Pro
2015-07-04 18:28:10,692 [main] INFO  org.apache.catalina.startup.Catalina - Server st
```

The application will be accessible at http://YOUR_IP:8080/openkm using user: **okmAdmin** with password: **admin**. (Do not forget to change it !). For more information: Manage users and roles.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Database

OpenKM stores most of its data in a SQL database. Several databases are supported, but they must be configured in order to work correctly.

Configure database

- Configuring MySQL and MariaDB
- Configuring Oracle
- Configuring PostgreSQL
- Configuring SQL Server
- Configuring other databases
- Configuring Hikari pool

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring MySQL and MariaDB

MariaDB is a community-developed split of the MySQSL intended to remain free under the GNU GPL. Being a split of a leading Open Source software system, it is notable for being led by the original developers of MySQL, who separated it due to concerns over its acquisition by Oracle.

Preliminaries

Perform all changes with the application stopped.

Check if your MySQL installation has InnoDB engine enabled:

```
$ mysql -h localhost -u root -p
> show engines;
```

It should show something like this:

| **InnoDB** | **DEFAULT** | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |

In case of another default engine, there are two options:

1. Use the **com.openkm.db.dialect.MySQL5InnoDBDialect** dialect and avoid changing the default MySQL Storage Engine.
2. Change default engine

Change default engine

Modify the MySQL configuration file named **my.cnf**

and under [mysqld] add

```
default-storage-engine = innodb
```

After changing the default to MySQL the service must be restarted to take effect.

Change default charset

In Ubuntu 16.04 the default charset configured in MariaDB and MySQL is **utf8mb4** but we recommend switching to **utf8**. In order to get the recommended charset you have to modify these files:

/etc/mysql/mariadb.conf.d/50-server.cnf

```
character-set-server = utf8
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
collation-server = utf8_general_ci
```

/etc/mysql/mariadb.conf.d/50-client.cnf

```
default-character-set = utf8
```

/etc/mysql/mariadb.conf.d/50-mysql-clients.cnf

```
default-character-set = utf8
```

Once modified, a database server restart is needed.

Database creation

Create a database and user

```
DROP DATABASE IF EXISTS okmdb;
CREATE DATABASE okmdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;
CREATE USER openkm@localhost IDENTIFIED BY '*secret*';
GRANT ALL ON okmdb.* TO openkm@localhost WITH GRANT OPTION;
```

You can check the database engine with:

```
$ mysqlshow -h localhost -u root -p --status okmdb;
```

More information at MySQL: Case Sensitivity in String Searches

Configure your openkm.properties

This configuration property should be set before the database creation. Once the database has been initialized don't modify it because can damage your installation.

If your OpenKM installation has been already configured with another database you can't switch to another database by simply changing this property.

Edit the file $TOMCAT_HOME/openkm.properties

```
spring.jpa.hibernate.ddl-auto=create-only
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect
```

MySQL Timezone

If you want to set your timezone in your jdbc connection according to your timezone configured in your system, first you can see the mappings both for MySQL and for MariaDB in this way:

```
SELECT * FROM mysql.time_zone_name
```

So, for example, you could configure your jdbc connection like this if your system timezone is for example UTC+1:

spring.datasource.url=jdbc:mysql://localhost:3306/okmdb?

autoReconnect=true&useUnicode=true&characterEncoding=UTF8&**serverTimezone=Europe/Madrid**

Where **Europe/Madrid** is a valid value taken from the table **time_zone_name.**

Configure Tomcat datasources

Edit the file $TOMCAT_HOME/openkm.properties and enable the resource named jdbc/OpenKMDS

MySQL:

```
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/okmdb?autoReconnect=true&useUnicode=
spring.datasource.username=openkm
spring.datasource.password=*secret*
```

MariaDB:

```
spring.datasource.driver-class-name=org.mariadb.jdbc.Driver
spring.datasource.url=jdbc:mariadb://localhost:3306/okmdb?autoReconnect=true&useUnicod
spring.datasource.username=openkm
spring.datasource.password=*secret*
```

MariaDB can use these driverClassName values:

- com.mysql.jdbc.Driver
- org.mariadb.jdbc.Driver

In recent MySQL versions, it seems you need to add **useSSL=false** to the connection URL.

Configures the attributes values named:

- userName.
- password.
- url (change to your hosts and port).

It can only use a resource named jdbc/OpenKMDS.

Configure application login

Edity the file $TOMCAT_HOME/openkm.properties

```
authentication.openkm.database=true
```

Check for JDBC Driver

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Go to $TOMCAT_HOME/lib and check for JDBC **mysql-connector-java-8.0.11.jar** Driver.

If it is not present, download MySQL JDBC driver from MySQL Home Page and move it.

The driver numeration version could be distinct due some version upgrade.

MariaDB can use its own JDBC mariadb-java-client-2.2.5.jar Driver.

If is not present, download mariaDB JDBC driver from MariaDB Connector/J and move it.

Run application

As a process into OpenKM startup, the application will create automatically and empty database structure.

Once the tables are created, OpenKM will automatically change the **hibernate.hbm2ddl** property from ***create-only*** *to* ***none***.

Additional information

- *InnoDB Performance Tuning Tips*
- *InnoDB - High disk write I/O on ibdata1 file and ib_logfile0*

Remove and create MySQL service in Windows

Ensure your mysql/bin folder is in your PATH at environment variables.

For example in MySQL 5.7 the bin folder location is at C:\Program Files\MySQL\MySQL Server 5.7\bin

If you have MySQL Window service you can remove with one of these command line:

```
c:\> mysqld --remove MySQL57
```

or

```
c:\> sc delete MySQL57
```

Create the service:

```
c:\> "C:\Program Files\MySQL\MySQL Server 5.7\bin\mysqld.exe" --install MYSQL57 --defaults-file="C:\ProgramData\MySQL\MySQL Server 5.7\my.ini"
```

More information at http://dev.mysql.com/doc/refman/5.7/en/windows-start-service.html

Migrate from Windows to Linux

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

By default table names in Windows are lowercase and in Linux in uppercase.

When you migrate from Windows to Linux you must convert lowercase table name to uppercase. The next script will helps on it:

```
select concat('rename table ', table_name, ' to ' , upper(table_name) , ';') from inf
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

MySQL tuning

Take this documentation as is, not as a definitive guide to tuning MySQL. There is a lot of information and knowledge that should be read before changing MySQL configuration. The documentation has been created based on experience shared by OpenKM server administrator users.

Useful links should read:

- *MySQL: InnoDB Performance Tuning Tips*
- *MySQL: The MySQL Query Cache*
- *InnoDB - High disk write I/O on ibdata1 file and ib_logfile0*

Sample one

Server information

- *Virtualized Windows Server 2008*
- *CPU: Intel Xeon X3470 @2,93GHz*
- *RAM: 7GB*
- *JVM settings for Tomcat: -Xms 3072 -Xmx 3072 -XX:PermSize=256m -XX:MaxPermSize=512m*
- *Repository is 1GB*

MySQL my.cnf changes

- *innodb_buffer_pool_size=1792M*
- *innodb_additional_mem_pool_size=16M*
- *innodb_log_file_size=256M*
- *innodb_log_buffer_size=8M*

Using MySQL server InnoDB engine with the default configuration parameters in my.cnf file can cause a serious performance bottleneck in applications with high database read write activity. MySQL should have enough RAM to be able to cache the repository and avoid the high Disk I/O activity.

The most important parameters are:

- ***innodb_buffer_pool_size***
- ***innodb_additional_mem_pool_size***
- ***innodb_log_file_size***
- ***innodb_log_buffer_size***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

According to some database experts, innodb_buffer_pool_size should occupy the 70 percent of RAM of a dedicated database server.

Before changing these values, please read MySQL documentation carefully. Before making any changes you should make sure that MySQL server has been stopped correctly (check server log), otherwise the innodb will become corrupted and MySQL server won't restart.

Considerations *Repository is 1GB so 1792M is enough RAM for* ***innodb_buffer_pool_size****. The larger the repository the more memory you need to allocate to* ***innodb_buffer_pool_size****. If you are using x86 MySQL server you cannot allocate RAM larger than 2GB.*

Information collected from OpenKM forum post.

Sample two

Server information

- *CPU: Intel Xeon E5606 @2.13GHz*
- *RAM: 4GB*
- *System: Windows server 2008, 64bit*
- *JAVA_OPTS=-Xms256m -Xmx1024m -XX:PermSize=256m -XX:MaxPermSize=512m -Djava.awt.headless=true -Dfile.encoding=utf-8*
- *Database size:2.5G*
- *Repository size:40GB*
- *100,000 nodes(folders + documents)*

MySQL my.cnf changes

- *query_cache_size=0*
- *table_open_cache=2000*
- *tmp_table_size=16M*
- *thread_cache_size=9*
- *myisam_max_sort_file_size=100G*
- *myisam_sort_buffer_size=32M*
- *key_buffer_size=8M*
- *read_buffer_size=64K*
- *read_rnd_buffer_size=256K*
- *sort_buffer_size=256K*
- *innodb_additional_mem_pool_size=16M*
- *innodb_log_buffer_size=8M*

- *innodb_buffer_pool_size=1042M*

Results

Before optimization the login time was about 45-55 seconds with a lot of hard-disk I/O usage. After the MySQL optimization login takes about 10 seconds.

With more RAM it will be possible to get a higher MySQL buffer and probably better perfomance.

Information collected OpenKM Forum post.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change MySQL data directory

Ubuntu

Stop the service:

```
$ service mysql stop
```

Create a new folder destination:

```
$ mkdir /new_path/mysql_data
$ cp -R -p /var/lib/mysql /new_path/mysql_data
```

MySQL user should have enough credentials granted to the new folder /new_path/mysql_data.

Optionally modify the path of older mysql data store folder location:

```
$ mv /var/lib/mysql /var/lib/mysql_old
```

*Edit the **datadir** property into the **/etc/mysql/my.cnf** configuration file:*

```
$ vim /etc/mysql/my.cnf
```

Other possible config paths:

/etc/mysql/mariadb.conf.d/50-server.cnf

/etc/mysql/mysql.conf.d/mysqld.cnf

Change the older data store folder path with the newer:

```
datadir = /var/lib/mysql
```

to

```
datadir = /new_path/mysql_data
```

*Enable appamor. Edit the file **/etc/apparmor.d/usr.sbin.mysqld**:*

```
$ vim /etc/apparmor.d/usr.sbin.mysqld
```

Change the older data store folder path with the newer:

```
/usr/lib/mysql/plugin/ r,
/usr/lib/mysql/plugin/*.so* mr,
...
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
/var/lib/mysql/ r,
/var/lib/mysql/** rwk,
```

to

```
/new_path/mysql_data/plugin/ r,
/new_path/mysql_data/plugin/*.so* mr,
...
/new_path/mysql_data/ r,
/new_path/mysql_data/** rwk,
```

Start MySQL service:

```
$ service mysql start
```

More information at http://www.ubuntugeek.com/how-to-change-the-mysql-data-default-directory.html.

CentOS and RHEL

Stop the service:

```
$ service mysqld stop
```

Create a new folder destination:

```
$ mkdir /new_path/mysql_data
$ cp -R -p /var/lib/mysql /new_path/mysql_data
```

The mysql user must have enough credentials granted to the new folder /new_path/mysql_data.

Optionally modify the path of older mysql data store folder location:

```
$ mv /var/lib/mysql /var/lib/mysql_old
```

*Edit the **datadir** property into the **/etc/mysql/my.cnf** configuration file:*

```
$ vim /etc/mysql/my.cnf
```

Change the older data store folder path with the newer:

```
datadir = /var/lib/mysql
```

to

```
datadir = /new_path/mysql_data
```

Enable appamor.

```
$ yum install policycoreutils-python
$ getenforce
$ semanage fcontext -a -t mysqld_db_t "/new_path/mysql_data(/.*)?"
$ restorecon -Rv /new_path/mysql_data
```

Start MySQL service:

```
$ service mysqld start
```

More information at http://lifeofageekadmin.com/how-to-change-the-default-mysql-data-directory-on-rhel-6/

Windows

Stop the service:

```
C:\> net stop mysql56
```

Create a new folder destination:

```
C:\> robocopy "C:\ProgramData\MySQL\MySQL Server 5.6\data" c:\new_path\mysql_data
```

Optionally modify the path of older mysql data store folder location:

```
C:\> ren "C:\ProgramData\MySQL\MySQL Server 5.6\data" "C:\ProgramData\MySQL\MySQL Server 5.6\data_old"
```

*Edit the **datadir** property into the **C:\ProgramData\MySQL\MySQL Server 5.6\my.ini** configuration file:*

Change the older data store folder path with the newer:

```
datadir = C:\ProgramData\MySQL\MySQL Server 5.6\data
```

to

```
datadir = c:\new_path\mysql_data
```

Start MySQL service:

```
C:\> net start mysql56
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

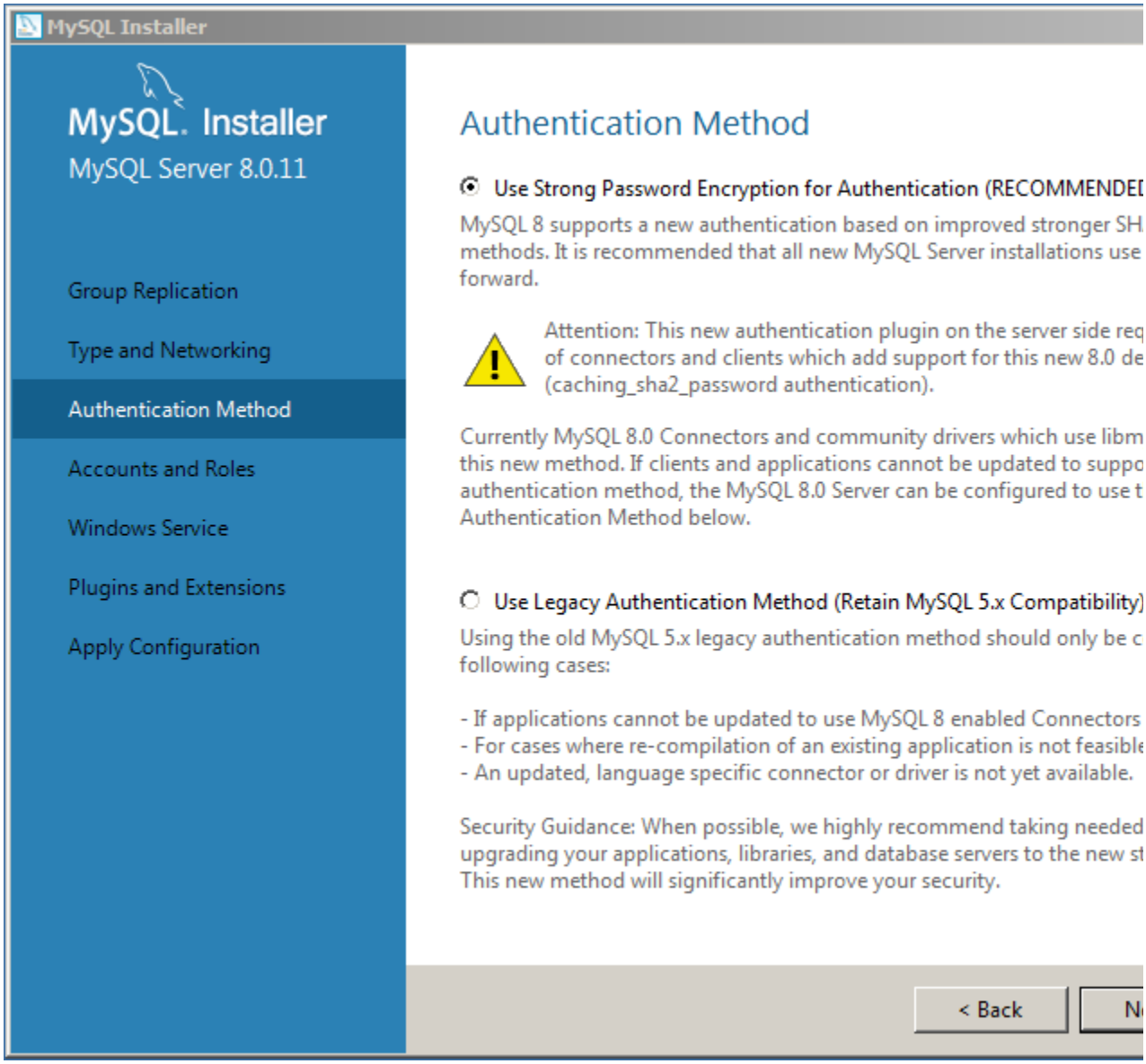
MySQL 8 troubleshooting

Could not create connection to database server

Might be several reason why the application is not able to stablish the connection to the database, take a look into the log file what might give to you the clues about what is happening.

Strong Password Encryption for Authentication

*The latest MySQL installation wizard suggest you to use "**Strong Password Encryption for Authentication**" as can be shown in the next screenshot:*



*When the option "**Strong Password Encryption for Authentication**" is enabled rather than "**Legacy Authentication Method**"*

*you are enabling **SSL connection by default**. Than means you might read some literature and apply some changes in your connection setting into your server.xml server to connect with the database. Usually in this scenario you will get some error like what is shown belown:*

```
java.sql.SQLException: Cannot create PoolableConnectionFactory (Could not create conn
 at org.apache.tomcat.dbcp.dbcp2.BasicDataSource.createPoolableConnectionFactory(Basi
 at org.apache.tomcat.dbcp.dbcp2.BasicDataSource.createDataSource(BasicDataSource.jav
 at org.apache.tomcat.dbcp.dbcp2.BasicDataSource.getConnection(BasicDataSource.java:1
 at org.hibernate.connection.DatasourceConnectionProvider.getConnection(DatasourceCon
 at org.hibernate.cfg.SettingsFactory.buildSettings(SettingsFactory.java:113) ~[hiber
 at org.hibernate.cfg.Configuration.buildSettingsInternal(Configuration.java:2863) [h
 at org.hibernate.cfg.Configuration.buildSettings(Configuration.java:2859) [hibernate
 at org.hibernate.cfg.Configuration.buildSessionFactory(Configuration.java:1870) [hib
 at com.openkm.dao.HibernateUtil.getSessionFactory(HibernateUtil.java:367) [classes/:
 at com.openkm.dao.HibernateUtil.getSessionFactory(HibernateUtil.java:182) [classes/:
 at com.openkm.servlet.RepositoryStartupServlet.init(RepositoryStartupServlet.java:99
 at javax.servlet.GenericServlet.init(GenericServlet.java:158) [servlet-api.jar:3.1.F
 at org.apache.catalina.core.StandardWrapper.initServlet(StandardWrapper.java:1183) [
 at org.apache.catalina.core.StandardWrapper.loadServlet(StandardWrapper.java:1099) [
 at org.apache.catalina.core.StandardWrapper.load(StandardWrapper.java:989) [catalina
 at org.apache.catalina.core.StandardContext.loadOnStartup(StandardContext.java:4931)
 at org.apache.catalina.core.StandardContext.startInternal(StandardContext.java:5241)
 at org.apache.catalina.util.LifecycleBase.start(LifecycleBase.java:150) [catalina.ja
 at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:752) [
 at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:728) [catalina
 at org.apache.catalina.core.StandardHost.addChild(StandardHost.java:734) [catalina.j
 at org.apache.catalina.startup.HostConfig.deployWAR(HostConfig.java:952) [catalina.j
 at org.apache.catalina.startup.HostConfig$DeployWar.run(HostConfig.java:1823) [catal
 at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511) [na:1.8.0
 at java.util.concurrent.FutureTask.run(FutureTask.java:266) [na:1.8.0_171]
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) [
 at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) [
 at java.lang.Thread.run(Thread.java:748) [na:1.8.0_171]
Caused by: java.sql.SQLNonTransientConnectionException: Could not create connection t
 at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:108) ~[mys
 at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:95) ~[mysq
 at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:87) ~[mysq
 at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:61) ~[mysq
 at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:71) ~[mysq
 at com.mysql.cj.jdbc.ConnectionImpl.connectWithRetries(ConnectionImpl.java:932) ~[my
 at com.mysql.cj.jdbc.ConnectionImpl.createNewIO(ConnectionImpl.java:857) ~[mysql-con
 at com.mysql.cj.jdbc.ConnectionImpl.<init>(ConnectionImpl.java:444) ~[mysql-connect
 at com.mysql.cj.jdbc.ConnectionImpl.getInstance(ConnectionImpl.java:230) ~[mysql-con
 at com.mysql.cj.jdbc.NonRegisteringDriver.connect(NonRegisteringDriver.java:226) ~[m
 at org.apache.tomcat.dbcp.dbcp2.DriverConnectionFactory.createConnection(DriverConne
 at org.apache.tomcat.dbcp.dbcp2.PoolableConnectionFactory.makeObject(PoolableConnect
 at org.apache.tomcat.dbcp.dbcp2.BasicDataSource.validateConnectionFactory(BasicDataS
 at org.apache.tomcat.dbcp.dbcp2.BasicDataSource.createPoolableConnectionFactory(Basi
 ... 27 common frames omitted
Caused by: com.mysql.cj.exceptions.CJCommunicationsException: Communications link fai

The last packet sent successfully to the server was 0 milliseconds ago. The driver ha
 at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method) ~[na:1.8.0_
 at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorIm
 at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAc
 at java.lang.reflect.Constructor.newInstance(Constructor.java:423) ~[na:1.8.0_171]
 at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:59
 at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:10
 at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:14
 at com.mysql.cj.exceptions.ExceptionFactory.createCommunicationsException(ExceptionF
 at com.mysql.cj.protocol.a.NativeProtocol.negotiateSSLConnection(NativeProtocol.java
 at com.mysql.cj.protocol.a.NativeAuthenticationProvider.negotiateSSLConnection(Nativ
 at com.mysql.cj.protocol.a.NativeAuthenticationProvider.proceedHandshakeWithPluggabl
 at com.mysql.cj.protocol.a.NativeAuthenticationProvider.connect(NativeAuthentication
```

```
 at com.mysql.cj.protocol.a.NativeProtocol.connect(NativeProtocol.java:1411) ~[mysql-
 at com.mysql.cj.NativeSession.connect(NativeSession.java:165) ~[mysql-connector-java
 at com.mysql.cj.jdbc.ConnectionImpl.connectWithRetries(ConnectionImpl.java:876) ~[my
 ... 35 common frames omitted
Caused by: javax.net.ssl.SSLHandshakeException: java.security.cert.CertificateExcepti
 at sun.security.ssl.Alerts.getSSLException(Alerts.java:192) ~[na:1.8.0_171]
 at sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1964) ~[na:1.8.0_171]
 at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:328) ~[na:1.8.0_171]
 at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:322) ~[na:1.8.0_171]
 at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1614) ~
 at sun.security.ssl.ClientHandshaker.processMessage(ClientHandshaker.java:216) ~[na:
 at sun.security.ssl.Handshaker.processLoop(Handshaker.java:1052) ~[na:1.8.0_171]
 at sun.security.ssl.Handshaker.process_record(Handshaker.java:987) ~[na:1.8.0_171]
 at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:1072) ~[na:1.8.0_171
 at sun.security.ssl.SSLSocketImpl.performInitialHandshake(SSLSocketImpl.java:1385) ~
 at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1413) ~[na:1.8.0_
 at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1397) ~[na:1.8.0_
 at com.mysql.cj.protocol.ExportControlled.performTlsHandshake(ExportControlled.java:
 at com.mysql.cj.protocol.StandardSocketFactory.performTlsHandshake(StandardSocketFac
 at com.mysql.cj.protocol.a.NativeSocketConnection.performTlsHandshake(NativeSocketCo
 at com.mysql.cj.protocol.a.NativeProtocol.negotiateSSLConnection(NativeProtocol.java
 ... 41 common frames omitted
Caused by: java.security.cert.CertificateException: java.security.cert.CertPathValida
 at com.mysql.cj.protocol.ExportControlled$X509TrustManagerWrapper.checkServerTrusted
 at sun.security.ssl.AbstractTrustManagerWrapper.checkServerTrusted(SSLContextImpl.ja
 at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:1596) ~
 ... 52 common frames omitted
Caused by: java.security.cert.CertPathValidatorException: Path does not chain with an
 at sun.security.provider.certpath.PKIXCertPathValidator.validate(PKIXCertPathValidat
 at sun.security.provider.certpath.PKIXCertPathValidator.engineValidate(PKIXCertPathV
 at java.security.cert.CertPathValidator.validate(CertPathValidator.java:292) ~[na:1.
 at com.mysql.cj.protocol.ExportControlled$X509TrustManagerWrapper.checkServerTrusted
 ... 54 common frames omitted
```

You have two options:

- *Configure SSL for the MySQL connection.*
- *Switch from "**Strong Password Encryption for Authentication**" to "**Legacy Authentication Method**".*

Some url what might be useful for you is you decide going ahead with SSL configuration:

- *https://dev.mysql.com/doc/connector-j/8.0/en/connector-j-reference-using-ssl.html*
- *https://dev.mysql.com/doc/refman/8.0/en/creating-ssl-rsa-files-using-mysql.html*

*In Windows is quite easy to enable "**Legacy Authentication Method**":*

- *Execute the installer.*
- *Click on reconfigure beside MySQL Server.*
- *Follow the wizard screens and choose "**Legacy Authentication Method**".*

Old JDBC driver

If in your log file is shown the error below:

```
Caused by: java.lang.NullPointerException: null
 at com.mysql.jdbc.ConnectionImpl.getServerCharset(ConnectionImpl.java:3005) ~[mysql-
 at com.mysql.jdbc.MysqlIO.sendConnectionAttributes(MysqlIO.java:1916) ~[mysql-connec
 at com.mysql.jdbc.MysqlIO.proceedHandshakeWithPluggableAuthentication(MysqlIO.java:1
 at com.mysql.jdbc.MysqlIO.doHandshake(MysqlIO.java:1215) ~[mysql-connector-java-5.1.
 at com.mysql.jdbc.ConnectionImpl.coreConnect(ConnectionImpl.java:2255) ~[mysql-conne
 at com.mysql.jdbc.ConnectionImpl.connectWithRetries(ConnectionImpl.java:2106) ~[mysq
 ... 41 common frames omitted
```

You are using a wrong driver version for connecting to the MySQL Server 8, you should make the next changes:

- *Replace the mysql-connector-java-5.1.38-bin.jar into your $TOMCAT_HOME/libs by the latest JDBC driver for MySQL Server 8. You can get the latest driver from https://dev.mysql.com/downloads/connector/j/.*
- *Change the MySQL connection settings set into your $TOMCAT_HOME/conf/server.xml, should be something like:*

*In the latest version of MySQL JDBC driver, the name of the class **com.mysql.jdbc.Driver** has been changed to **com.mysql.cj.jdbc.Driver**.*

```
<Resource name="jdbc/OpenKMDS" auth="Container" type="javax.sql.DataSource"
            maxTotal="100" maxIdle="30" maxWaitMillis="10000" validationQuery="select
            username="openkm" password="openkm" driverClassName="com.mysql.cj.jdbc.Dr
            url="jdbc:mysql://localhost:3306/okmdb?autoReconnect=true&amp;useUnicode=
```

The server time zone value is unrecognized

If in your log file is shown the error below or similar:

```
Caused by: com.mysql.cj.exceptions.InvalidConnectionAttributeException: The server ti
 at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method) ~[na:1.8.0_
 at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorIm
 at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAc
 at java.lang.reflect.Constructor.newInstance(Constructor.java:423) ~[na:1.8.0_171]
 at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:59
 at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:83
 at com.mysql.cj.util.TimeUtil.getCanonicalTimezone(TimeUtil.java:128) ~[mysql-connec
 at com.mysql.cj.protocol.a.NativeProtocol.configureTimezone(NativeProtocol.java:2201
 at com.mysql.cj.protocol.a.NativeProtocol.initServerSession(NativeProtocol.java:2225
 at com.mysql.cj.jdbc.ConnectionImpl.initializePropsFromServer(ConnectionImpl.java:13
 at com.mysql.cj.jdbc.ConnectionImpl.connectWithRetries(ConnectionImpl.java:895) ~[my
 ... 45 common frames omitted
```

*It seems the latest MySQL JDBC driver have some issue with your OS time zone configuration, a quick workaround is to indicate the server time zone, for example **serverTimezone=UTC**:*

```
<Resource name="jdbc/OpenKMDS" auth="Container" type="javax.sql.DataSource"
            maxTotal="100" maxIdle="30" maxWaitMillis="10000" validationQuery="select
            username="openkm" password="openkm" driverClassName="com.mysql.cj.jdbc.Dr
            url="jdbc:mysql://localhost:3306/okmdb_test?autoReconnect=true&amp;useUni
```

Problem connecting to your database

If you get an error like this:

```
Caused by: com.mysql.jdbc.exceptions.jdbc4.MySQLNonTransientConnectionException: Publ
```

You need to modify your file ***conf/server.xml*** *and add to URL by JDBC the follow:* ***allowPublicKeyRetrieval=true***.

```
<Resource name="jdbc/OpenKMDS" auth="Container" type="javax.sql.DataSource"
            maxTotal="100" maxIdle="30" maxWaitMillis="10000" validationQuery="select
            username="openkm" password="openkm" driverClassName="com.mysql.cj.jdbc.Dr
            url="jdbc:mysql://localhost:3306/okmdb_test?allowPublicKeyRetrieval=true&
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Oracle

Database creation

Create tablespace

```
create tablespace OPENKMDATA datafile '/u01/app/oracle/oradata/XE/openkmdata.dbf' siz
```

Create a database and user

```
create user OPENKM identified by *secret* default tablespace OPENKMDATA quota unlimit
grant all privileges to OPENKM;
grant create any context, create cluster, create dimension, create indextype, create
```

Configure your openkm.properties

This configuration property should be set before the database creation. Once the database has been initialized don't modify it because can damage your installation.

If your OpenKM installation has been already configured with another database you can't switch to another database simply changing this property.

Choose the dialect what fits better with your database version:

- *Oracle10gDialect*
- *Oracle12cDialect*
- *Oracle8iDialect*
- *Oracle9iDialect*
- *Oracle9Dialect: deprecated, use Oracle9iDialect or Oracle10gDialect*
- *OracleDialect: deprecated, use Oracle8iDialect*

Edit the file $TOMCAT_HOME/`openkm.properties`

```
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.Oracle10gDialect
spring.jpa.hibernate.ddl-auto=create-only
```

Configure Tomcat datasources

Edit the file $TOMCAT_HOME/openkm.properties and enable the resource named jdbc/OpenKMDS

```
spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE
spring.datasource.username=OPENKM
spring.datasource.password=*secret*
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
spring.datasource.validationQuery=select 1 from dual
```

Sample using service name instead of SID:

The main difference between SID connection is the ":" is replaced by "/".

```
spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
spring.datasource.url=jdbc:oracle:thin:@localhost:1521/ServiceName
spring.datasource.username=OPENKM
spring.datasource.password=*secret*
spring.datasource.validationQuery=select 1 from dual
```

Another sample:

```
spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
spring.datasource.url=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(HOST=localhost)(PROTOC
spring.datasource.username=OPENKM
spring.datasource.password=*secret*
spring.datasource.validationQuery=select 1 from dual
```

Configure the attributes values named:

- *userName.*
- *password.*
- *url (change to your hosts and port).*

If you find this error starting the tomcat:

ORA-01882: timezone region not found

*Put this line in the file **$CATALINA_BASE/bin/setenv.sh**:*

```
JAVA_OPTS=-Doracle.jdbc.timezoneAsRegion=false
```

Configure application login

Edity the file $TOMCAT_HOME/openkm.properties

```
authentication.openkm.database=true
```

Run application

As a process into OpenKM startup, the application will create automatically and empty database structure.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Once the tables are created, OpenKM will automatically change the ***hibernate.hbm2ddl*** *property from* ***create-only*** *to* ***none****.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring PostgreSQL

Database creation

Create a database and user

```
DROP DATABASE IF EXISTS okmdb;
CREATE USER openkm WITH PASSWORD '*secret*';
CREATE DATABASE okmdb WITH OWNER openkm ENCODING 'UTF8';
```

or from the command line

```
$ createuser --pwprompt openkm
$ createdb --owner=openkm --encoding=UTF8 okmdb
```

More info at:

- *PostgreSQL 9.1 Documentation*
- *Guía Ubuntu: PostgreSQL*

Configure your openkm.properties

This configuration property should be set before the database creation. Once the database has been initialized don't modify it because can damage your installation.

If your OpenKM installation has been already configured with another database you can't switch to another database simply changing this property.

Edit the file $TOMCAT_HOME/openkm.properties

```
hibernate.dialect=com.openkm.db.dialect.PostgreSQL9Dialect
hibernate.hbm2ddl=create-only
```

Configure Tomcat datasources

Edit the file $TOMCAT_HOME/openkm.properties and enable the resource named jdbc/OpenKMDS

```
spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://localhost:5432/okmdb
spring.datasource.username=openkm
spring.datasource.password=*secret*
spring.datasource.validationQuery=SELECT 1
```

Configure the attributes values named:

- *userName.*

- *password.*
- *url (change to your hosts and port).*

Configure application login

Edity the file $TOMCAT_HOME/openkm.properties

```
authentication.openkm.database=true
```

Enable native configuration parameter

Go to Administration > Configuration parameters

```
native.sql.lowercase=true
```

Run application

As a process into OpenKM startup, the application will create automatically and empty database structure.

*Once the tables are created, OpenKM will automatically change the **hibernate.hbm2ddl** property from **create-only** to **none**.*

Additional information

If you want to obtain the maximun performance from PostgreSQL, pgFouine will help in your task.

PostgreSQL 9.6 and upper

*From **PostgreSQL version 9.6 and upper**, has been **changed some tables and columns of the schema**. Check that you are using the right JDBC driver.*

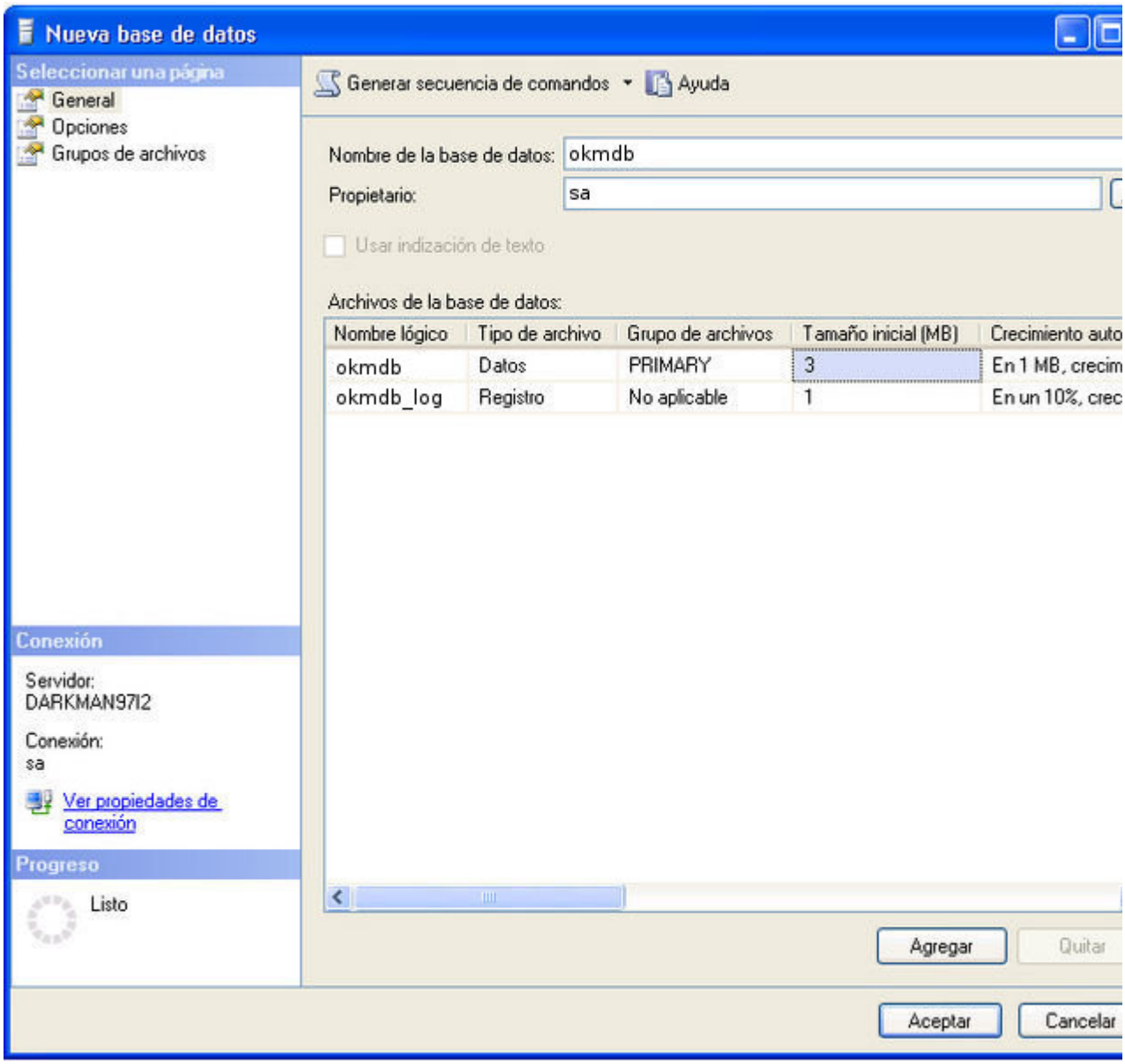
Read this related issue for more information PostgreSQL - ERROR o.h.tool.hbm2ddl.SchemaUpdate - could not complete schema update

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

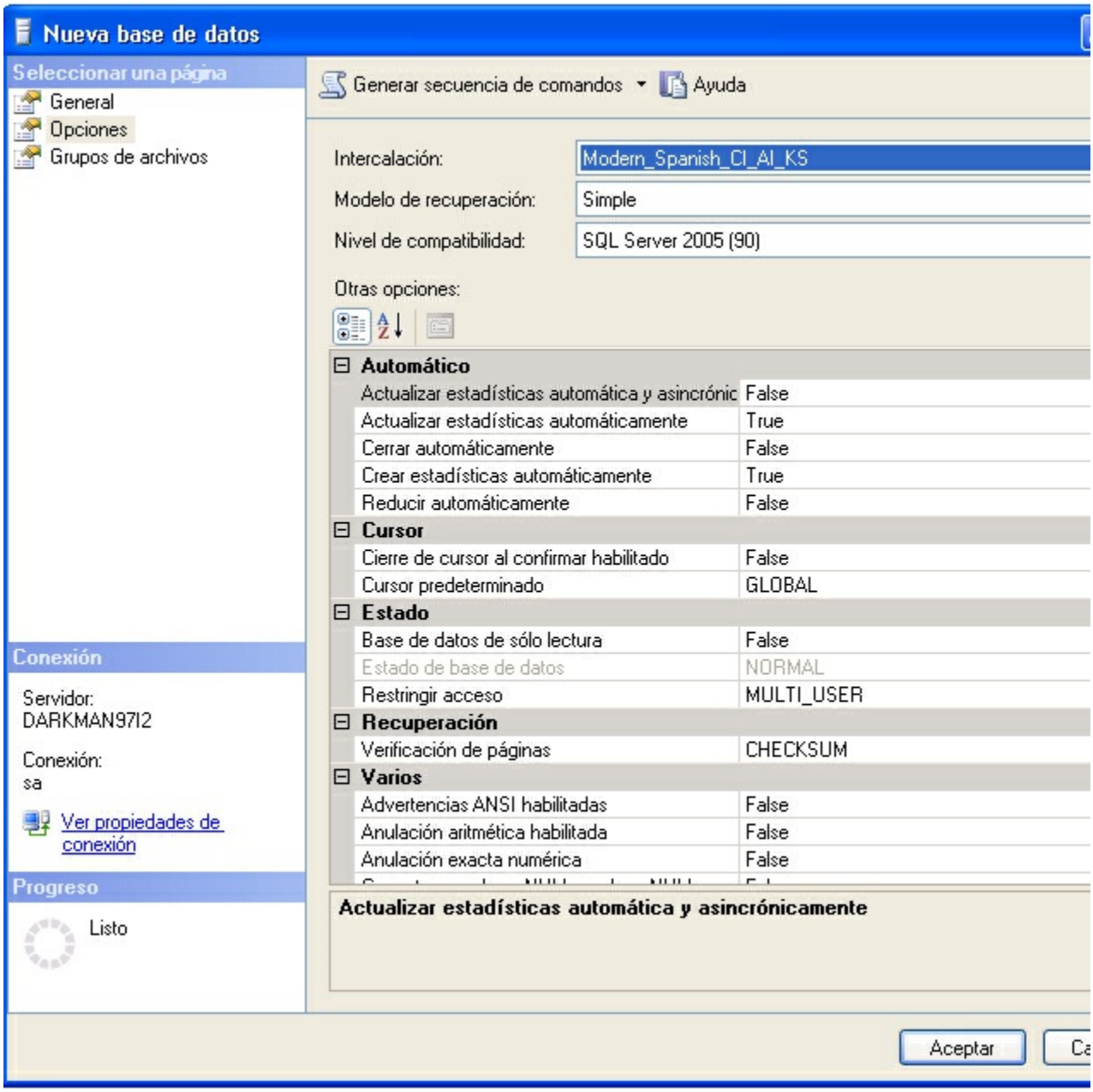
Database creation

Create a database and user

Start SQL Server Management Studio Express and create a database called okmdb.



Select your appropriate database collate.



*We checked with **Modern_Spanish_CI_AI_KS**, where **CI indicate case-insensitive** collation and **AS indicate accent-sensitive** collation.*

*To ensure the correct OpenKM functionality it should be selected **case-insensitive***

Configure your openkm.properties

This configuration property should be set before the database creation. Once the database has been initialized don't modify it because can damage your installation.

If your OpenKM installation has been already configured with another database you can't switch to another database simply changing this property.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Choose the hibernate dialect what fits better with your current SQLServer database version:

- *SQLServer2005Dialect*
- *SQLServer2008Dialect*
- *SQLServer2012Dialect*
- *SQLServerDialect*

Edit the file $TOMCAT_HOME/openkm.properties

```
spring.jpa.hibernate.ddl-auto=create-only
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.SQLServerDialect
```

Configure Tomcat data sources

Edit the file $TOMCAT_HOME/openkm.properties and enable the resources

```
spring.datasource.driver-class-name=com.microsoft.sqlserver.jdbc.SQLServerDriver
spring.datasource.url=jdbc:sqlserver://localhost:1433;databaseName=okmdb
spring.datasource.username=openkm
spring.datasource.password=*secret*
spring.datasource.validationQuery=SELECT 1
```

Configure the attributes values named:

- *userName.*
- *password.*
- *URL (change to your hosts and port).*

Configure application login

Edit the file $TOMCAT_HOME/openkm.xml

```
authentication.openkm.database=true
```

Run application

As a process into the OpenKM startup, the application will create automatically an empty database structure.

*Once the tables are created, OpenKM will automatically change the **hibernate.hbm2ddl** property from **create-only** to **none**.*

Troubleshooting error:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

You can not connect to MS SQL Server

Ensure you've got TCP protocol enabled (by default is disabled)

Connection lost after some time

We have observed in some SQL Server editions what seems a new behaviour what passed some hours the connection is closed (might be some extra configuration parameter in the latest SQL server editions what comes enabled by default or similar).

There's a new configuration attribute name ***autoReconnect*** *what solves it.*

```
spring.datasource.driver-class-name=com.microsoft.sqlserver.jdbc.SQLServerDriver
spring.datasource.url=jdbc:sqlserver://localhost:1433;databaseName=okmdb;autoReconnec
spring.datasource.username=openkm
spring.datasource.password=*secret*
spring.datasource.validationQuery=SELECT 1
```

Not able to login into OpenKM

When first-time OpenKM login it fails with right username and password (***okmAdmin*** */* ***admin)****, you should check database configuration.*

Check if the database is configured to be ***case-sensitive****, might be one of the reasons why you are not able to login.*

If you have this troubleshooting, you should to drop the database and create again selecting the collation as ***case-insensitive***

Not able to connect TCP/ IP to localhostSQL, port 1433 when use instance in the JDBC connection URL

In ***case you use database instance name*** *in the connection you must use "\\" in the* ***spring.datasource.url*** *configuration parameter. The sample below it use* ***database intance named SampleSQL****.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
spring.datasource.url=jdbc:sqlserver://localhost\\SampleSQL;databaseName=okmdb;autoRe
```

Additional information

- *How to enable remote connections in SQL Server 2008*
- *How do I configure SQL Server 2005/2008 to accept SQL Authentication*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring other databases

OpenKM uses Hibernate that means can be installed on major DBMS. For more information take a look at Hibernate: SQL Dialects.

By default application has been tested in:

- *MySQL or MariaDB.*
- *PostgreSQL.*
- *MS SQL Server.*
- *Oracle.*

If you want to check other database, contact with your assigned OpenKM technical staff guys at OpenKM customer support website.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Hikari pool

OpenKM use Hikari pool, the default configuration parameters are:

```
spring.datasource.hikari.minimumIdle=5
spring.datasource.hikari.maximumPoolSize=50
spring.datasource.hikari.idleTimeout=30000
spring.datasource.hikari.maxLifetime=1800000
spring.datasource.hikari.connectionTimeout=30000
spring.datasource.hikari.leak-detection-threshold=48000
```

You can modify these values setting your own in the $TOMCAT_HOME/openkm.properties file.

Enable debug

```
logging.level.com.zaxxer.hikari.pool.HikariPool=DEBUG
logging.level.com.zaxxer.hikari.HikariConfig=DEBUG
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Backup and restore

One of the most important things to have is an updated backup of your OpenKM.

If you have followed the installation instructions then you have configured OpenKM to use a production-safe database and optionally you have set a path the OpenKM repository data store.

OpenKM has several configurations but for optimal performance we recommend:

- *Use default data store what saves binary information directly to the server disk (in a non human comprehensible format).*
- *All the other information, folders and documents structure, metadata, etc., is stored into database.*

OpenKM allows to use also other datastore, for example to save binary data into a database.

In that case:

1. *First backup your database (make a SQL dump)*
2. *Then backup your data on filesystem*

*By default, all documents are stored in $TOMCAT_HOME/repository. This default folder can be changed using the property "**repository.home**" in the file $TOMCAT_HOME/openkm.properties file. Remember from OpenKM version 7.1.x the **lucene indexes path location** in the file system **are not managed by "repository.home"**. For more information take a look at Change repository home.*

Although OpenKM supports hot backups of your data, it is recommended to stop Tomcat before starting the backup. Apply this good practice when possible.

Backup steps:

- *Stop Tomcat (optional).*
- *Backup the database (mandatory).*
- *Backup the repository folder (mandatory).*
- *Start Tomcat (optional).*

Performing the backup in that order (the database first, then the filesystem) will ensure backup consistency.

There is an alternative to traditional backup methods: you can export the whole repository from OpenKM Administration > Utilities > Repository export and choose metadata and history options. But we do not suggest this way of doing backups. For medium to large repositories we suggest some incremental backup solution.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Detailed steps

In case you have OpenKM configured with MySQL or MariaDB you can follow these steps to backup and restore. For other databases check its corresponding command.

Backup

```
$ /etc/init.d/tomcat stop
$ mysqldump -h localhost -u root -p okmdb > /path/to/openkm/okmdb.sql
$ zip -r /path/to/backup/okm-backup.zip /path/to/openkm
$ /etc/init.d/tomcat start
```

After that, you will have a ZIP archive with the whole OpenKM installation and the exported database. Please, check before destroying this installation. We always recommend to restore the backup to ensure you have all the required files.

Restore

Please, note that you have to create the database if wasn't created previously. In case of MySQL or MariaDB and Ubuntu, use this command to install it:

```
$ sudo apt install mariadb-server
```

In case of CentOS:

```
$ sudo yum install mariadb-server
```

And execute the following sentences:

```
CREATE USER 'openkm'@'localhost' IDENTIFIED BY 's3cr3t0';
CREATE DATABASE okmdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_bin;
GRANT ALL ON okmdb.* TO 'openkm'@'localhost' WITH GRANT OPTION;
```

Keep on mind that the user and database name may be different in you case. Please, take a look at the installation report or check it at $TOMCAT_HOME/openkm.properties

Once created restore the database and files:

```
$ unzip /path/to/backup/okm-backup.zip -d /path/to/restore
$ mysql -h localhost -u root -p okmdb < /path/to/restore/okmdb.sql
```

Set the permissions according to the user who start OpenKM:

```
$ sudo chown openkm:openkm /path/to/openkm -R
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Install the required dependencies. In case of Ubuntu:

```
$ sudo apt install gawk imagemagick libreoffice tesseract-ocr
```

In case of CentOS:

```
$ sudo yum install gawk ImageMagick libreoffice libreoffice-headless
```

Once restored, verify file permissions, start OpenKM and make a repository check. See *Repository checker*.

Backup scripts

- *Linux backup and restore with BorgBackup* ***(recommended)***
- *Linux backup and restore with rdiff-backup tool*
- *Linux backup and restore with rsync*
- *Linux backup and restore with Duplicity*
- *Linux backup with LFTP*

Additional information

- *Easy Automated Snapshot-Style Backups with Linux and Rsync*
- *BorgBackup documentation*
- *Rdiff-backup home page*
- *Duplicity Backup Howto*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux backup and restore with BorgBackup

Borg is a deduplicating backup program which also supports compression and authenticated encryption. It provides an efficient and secure way to backup data. The data deduplication technique used makes Borg suitable for daily backups since only changes are stored. The authenticated encryption technique makes it suitable for backups to not fully trusted targets.

A more detailed description of BorgBackup can be found at https://borgbackup.readthedocs.io/en/stable/index.html.

Installation

Borg is included in many Linux distributions repositories but sometimes the packaged version is quite old, and it is better to install manually. This installation is as simple as:

> *In the official borg releases website section you will find the latest version:*
>
> *https://github.com/borgbackup/borg/releases*

```
$ wget https://github.com/borgbackup/borg/releases/download/1.1.5/borg-linux64
$ chmod +x borg-linux64
$ mv borg-linux64 /usr/local/bin/borg
```

Sample script

Backup to remote computer with SSH

```
#!/bin/bash
#
## BEGIN CONFIG ##
HOST=$(uname -n)
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_HOST="backup@bk-server.com"
BACKUP_DIR="/mnt/backup/$HOST"
BACKUP_REPO="${BACKUP_HOST}:${BACKUP_DIR}"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

echo "Backup destination ${BACKUP_REPO}"
if (ssh ${BACKUP_HOST} "[ ! -d ${BACKUP_DIR} ]"); then
  echo "Initialize backup repository: ${BACKUP_DIR}"
  ssh ${BACKUP_HOST} "mkdir -p ${BACKUP_DIR}"
  borg init -e none ${BACKUP_REPO}
fi

# Database
echo "Backup MySQL databases"
mkdir -p ${DATABASE_EXP}
echo "- Backuping $OPENKM_DB..."
mysqldump --single-transaction --add-drop-table --quick ${OPENKM_DB} > ${DATABASE_EXP
```

```
borg create -v --stats ${BACKUP_REPO}::'{now:%Y-%m-%d_%H:%M}' ${OPENKM_HOME}
borg prune -v ${BACKUP_REPO} --list --keep-daily=7 --keep-weekly=4 --keep-monthly=6
```

This is an improved SSH version

```
#!/bin/bash

# Configuration
DATABASE_EXP="/home/openkm/db"
BACKUP_HOST="backup@bk-server.com"
BACKUP_DIR="/mnt/backup/$HOST"
BACKUP_REPO="${BACKUP_HOST}:${BACKUP_DIR}"

echo "### BEGIN: $(date +"%x %X") ###"
echo "Backup destination ${BACKUP_REPO}"
TBEGIN=$(date +"%s")

# Check if the repository is already initialized
borg list ${BACKUP_REPO} > /dev/null
RESULT=$?; if [[ ${RESULT} != 0 ]]; then
  echo "Initializing backup repository..."
  borg init -e none ${BACKUP_REPO}
fi

# Database
echo "Backup MySQL databases"
mkdir -p ${DATABASE_EXP}
MYSQL_DBS=$(mysqlshow | awk '(NR > 2) && (/[a-zA-Z0-9]+[ ]+[|]/) { print $2 }');
for DB in $MYSQL_DBS; do
  if [[ $DB != "mysql" && $DB != "test" && $DB != "information_schema" && $DB != "per
    echo "- Backuping $DB..."
    mysqldump --single-transaction --add-drop-table --quick ${DB} > ${DATABASE_EXP}/m
  fi
done

borg create -v --stats ${BACKUP_REPO}::'{now:%Y-%m-%d_%H:%M}' /opt /var /etc /root /h
RESULT=$?; if [[ ${RESULT} != 0 ]]; then
  echo "**************************************************"
  echo " Backup error => exit code: ${RESULT}"
  echo "**************************************************"
fi

# Prune old backups
echo "Prune old backups"
borg prune -v --list --keep-within=30d ${BACKUP_REPO}
RESULT=$?; if [[ ${RESULT} != 0 ]]; then
  echo "**************************************************"
  echo " Backup error => exit code: ${RESULT}"
  echo "**************************************************"
fi

TEND=$(date +"%s")
TDIFF=$(($TEND-$TBEGIN))
echo "### END: $(date +"%x %X") - $(($TDIFF / 60)) mins and $(($TDIFF % 60)) secs ###
```

Backup to mapped network

```
#!/bin/bash
#
## BEGIN CONFIG ##
HOST=$(uname -n)
```

```
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_DIR="/mnt/backup"
BACKUP_REPO="${BACKUP_DIR}/${HOST}"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  mount "$BACKUP_DIR";

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "$BACKUP_DIR error mounting";
    exit -1;
  fi
fi

# Initialize repository
echo "Backup destination ${BACKUP_REPO}"
if ! ls -l ${BACKUP_DIR} | grep '^d' | grep "${HOST}" > /dev/null; then
  echo "Initialize backup repository: ${BACKUP_REPO}"
  mkdir ${BACKUP_REPO}
  borg init -e none ${BACKUP_REPO}
fi

# Database
echo "Backup MySQL databases"
mkdir -p ${DATABASE_EXP}
echo "- Backuping $OPENKM_DB..."
mysqldump --single-transaction --add-drop-table --quick ${OPENKM_DB} > ${DATABASE_EXP

# Data backup
echo "File system backup: ${BACKUP_REPO}"
borg create -v --stats ${BACKUP_REPO}::'{now:%Y-%m-%d_%H:%M}' ${OPENKM_HOME}
borg prune -v ${BACKUP_REPO} --list --keep-daily=7 --keep-weekly=4 --keep-monthly=6

# Umount disk
sync
umount "$BACKUP_DIR"
echo "$BACKUP_DIR umounted";
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs

$ sudo crontab -e
```

And add these lines according to your personal configuration:

```
MAILTO=nomail@openkm.com
PATH=/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/bin
@daily /root/backup.sh 2>&1 | tee /root/logs/backup.$(date +\%Y.\%m.\%d_\%H.\%M.\%S).
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

Recovery

*In order to list all stored backup use the **list** action parameter:*

```
$ borg list backup@bk-server.com:/mnt/backup/name
2018-02-07 Thu, 2018-02-07 10:12:31 [c878b314107c76b2d86ba75a9e5b2a216bf1f0d6b824313396c51ccda02a72e6]
2018-02-08 Thu, 2018-02-08 10:15:45 [2cf11f13bc7a2c9c0f1a0c243da52ac4b7657e7c6429cca0bf06a80467837f1e]
```

If you want to list all files in a given backup:

```
$ borg list backup@bk-server.com:/mnt/backup/name::2018-02-07_09:27
```

In case you need to access the contents of a backup you can mount as a local filesystem:

```
$ mkdir /tmp/backup
$ borg mount backup@bk-server.com:/mnt/backup/name::2018-02-07_09:27 /tmp/backup
```

If you do not specify an specific backup then will be mounted all the backup. The command line should be:

```
$ mkdir /tmp/backup
$ borg mount backup@bk-server.com:/mnt/backup /tmp/backup
```

At the end you might be interested in umount the folder:

```
borg umount /tmp/backup
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux backup and restore with rdiff-backup tool

Rdiff-backup backs up one directory to another, possibly over a network. The target directory ends up a copy of the source directory, but extra reverse diffs are stored in a special subdirectory of that target directory, so you can still recover files lost some time ago.

The idea is to combine the best features of a mirror and an incremental backup. rdiff-backup also preserves subdirectories, hard links, dev files, permissions, uid/gid ownership, modification times, extended attributes, acls, and resource forks.

Also, rdiff-backup can operate in a bandwidth efficient manner over a pipe, like rsync. Thus you can use rdiff-backup and ssh to securely back a hard drive up to a remote location, and only the differences will be transmitted.

> *If you need to backup to CIFS (SMB, Samba) or Mac's HFS, please take a look at rdiff-backup FAQ.*

Debian & Ubuntu

This utility is included in the software repository, so to install just run as root:

```
$ apt-get install rdiff-backup
```

> *Rdiff-backup from Debian issues a warning message due to a deprecated method [os.popen2 is deprecated], which can be hidden following the steps at How to shut up Python deprecation warnings.*

CentOS & RedHat

In this case, the application is not included in the default software repository so you need to make use of the EPEL repository:

```
$ rpm -Uvh http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-5.noarch.rpm
$ yum update
$ yum install rdiff-backup
```

Preliminaries

In these scripts are assumed these default values:

- ***$TOMCAT_HOME** value is "**/home/openkm/tomcat-7.0.61**"*
- *OpenKM **database** is named "**okmdb**".*
- *OpenKM **database password** value is "***secret***".*
- *OpenKM start & stop **service script** location is "**/etc/init.d/tomcat**"*

Sample script

As a good practice the backup should be done by root user.

The script below does a backup to USB and the application database is MySQL.

USB disk mount point can be defined in /etc/fstab as:

```
/dev/sdb1    /mnt/backup    ext4    defaults    0    0
```

These are the global script sections:

- *Global var configuration under **## BEGIN CONFIG ##** section.*
- *Checking user root under **# Check root** user section.*
- *Deleting older database backup **# Delete older local database backup** section.*
- *Mount disk (optional section) under **# Mount disk** section.*
- *Stop application (optional section) under **# Stop tomcat** section.*
- *Clean logs (optional section) under **# Clean logs** section.*
- *Backup the database to the filesystem under **#Backup database** section.*
- *Backup repository and database dump to backup destination under **#Backup and purge old backups** section.*
- *Start application (optional section) **# Start tomcat** section.*
- *Show statistics (optional section) **# Status** section.*
- *Umount disk (optional section) under **# Umount disk** section.*

Configuration parameters explanation:

Parameter	*Description*
HOST	*The server host name.*
DATABASE_PASS	*The database password.* *The datase user to do backup is always root user.*
OPENKM_DB	*The application database name.* *Usually will be named "okmdb".*

OPENKM_HOME	*The openkm home folder.*
TOMCAT_HOME	*The tomcat home folder.* *Usually into OPENKM_HOME folder.*
DATABASE_EXP	*The database dump folder.* *Usually into the same server.*
BACKUP_DIR	*Backup destination. In case of a remote server, the form is user@server::/path/to/backup*

*Create a backup script **/root/backup.sh***

```
$ sudo su
$ vim /root/backup.sh
```

```
#!/bin/bash
#
## BEGIN CONFIG ##
HOST=$(uname -n)
DATABASE_PASS="*secret*"
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat-7.0.61"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_DIR="/mnt/backup"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

# Delete older local database backup
echo -e "### BEGIN: $(date +"%x %X") ###\n"
rm -rf $DATABASE_EXP
mkdir -p $DATABASE_EXP

# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  mount "$BACKUP_DIR";

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "$BACKUP_DIR error mounting";
    exit -1;
  fi
fi
```

```
# Stop Tomcat
/etc/init.d/tomcat stop

# Clean logs
#echo "Clean Tomcat temporal files."
#rm -rf $TOMCAT_HOME/logs/*
#rm -rf $TOMCAT_HOME/temp/*
#rm -rf $TOMCAT_HOME/work/Catalina/localhost

# Backup database
if [ -n "$DATABASE_PASS" ]; then
  echo "* Backuping MySQL data from $OPENKM_DB..."
  mysqldump -h localhost -u root -p$DATABASE_PASS $OPENKM_DB > $DATABASE_EXP/mysql_$O
  echo "-------------------------------------";
fi

# Backup and purge old backups
rdiff-backup --remove-older-than 30B $BACKUP_DIR/$HOST
rdiff-backup -v 3 --print-statistics --include $OPENKM_HOME --exclude '**' / $BACKUP_

# Start Tomcat
/etc/init.d/tomcat start
echo -e "\n### END: $(date +"%x %X") ###"

# Status
echo "=================================";
rdiff-backup --list-increment-sizes $BACKUP_DIR/$HOST
echo "*********************************";
df -h | grep "$BACKUP_DIR"
echo "=================================";

# Umount disk
sync
umount "$BACKUP_DIR"
```

Increase the backup period

*Modify the **# Backup and purge old backups** section increasing the parameter value:*

> *The "**30B**" value indicate that will be able to restore a full backup of any of the last 30 days, increasing to " **90B**" will indicate any of the last 90 days, etc...*

```
--remove-older-than 90B
```

Do remote backup

*Modify the **# Backup and purge old backups** section.*

> *The backup@server indicate remote "server" accessed by user "backup".*

```
# Backup and purge old backups
rdiff-backup --remove-older-than 30B backup@server::$BACKUP_DIR/$HOST
rdiff-backup -v 3 --print-statistics --include $FILES --exclude '**' / backup@server:
```

*Also could be only modified the **## Config section**, for example:*

```
## BEGIN CONFIG ##
BACKUP_DIR="user@server::/path/to/backup"
```

*And keep the **# Backup and purge old backups** intact.*

```
# Backup and purge old backups
rdiff-backup --remove-older-than 30B $BACKUP_DIR/$HOST
```

Do remote backup to SMB or CIFS

*Modify the **# Mount disk** section.*

```
# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  echo "Mounting $BACKUP_DIR ...";
  mount -t cifs //REMOTE_SERVER/REMOTE_DIR $BACKUP_DIR -o username=REMOTE_USER,passwo

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "Error mounting $BACKUP_DIR.";
    exit -1;
  fi
fi
```

Do PostgreSQL backup

*Modify the **# Backup database** section*

```
# Backup de PostgreSQL
echo "* Backuping PostgreSQL data from $OPENKM_DB..."
su postgres -c "pg_dump $OPENKM_DB" > $DATABASE_EXP/pg_$OPENKM_DB.sql

# Databases optimizations
su postgres -c "vacuumdb -a -z" > /dev/null
su postgres -c "reindexdb -a -q" 2> /dev/null
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs

$ sudo crontab -e
```

And add these lines according to your personal configuration:

```
MAILTO=nomail@openkm.com
@daily /root/backup.sh | tee /root/logs/backup.$(date +\%Y.\%m.\%d_\%H.\%M.\%S).log
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

Restoring backup

You can list the available backups:

```
$ rdiff-backup --list-increment-sizes /path/to/backup
Sun Sep 11 05:00:19 2012        4.25 GB         4.25 GB   (current mirror)
Sun Sep  4 00:00:18 2012        13.3 MB         4.26 GB
Sun Aug 28 00:00:13 2012         674 MB         4.92 GB
Sun Aug 21 00:00:14 2012        5.50 MB         4.93 GB
Sun Aug 14 00:00:16 2012        1.75 MB         4.93 GB
Sun Aug  7 00:00:12 2012         288 KB         4.93 GB
Sun Jul 31 00:00:13 2012        43.0 KB         4.93 GB
Fri Jul 29 10:36:39 2012        5.56 KB         4.93 GB
```

Then decide to restore some of the backups, for example the one made on Sun Aug 28 00:00:13 2012:

```
$ rdiff-backup --restore-as-of 2012-08-28 /path/to/backup /path/to/destination
```

Or restore the last backup

```
$ rdiff-backup --restore-as-of now /path/to/backup /path/to/destination
```

*The **--restore-as-of** parameter accepts several formats. See **rdiff-backup** documentation for more info.*

Sample for restoring single folder (the /home/openkm/tomcat-7.0.61/ must exists)

```
rdiff-backup --restore-as-of now /mnt/backup/ms1/home/openkm/tomcat-7.0.61/ /home/openkm/tomcat-7.0.61/
```

*Inside of **/path/to/destination** you should see a directory **/home/openkm**, and inside it a couple of directories:*

- ***db**: The backup of the database.*
- ***tomcat-7.0.61**: The backup of the Tomcat installation and OpenKM repository into.*

Additional information

- *rdiff-backup home page*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *rdiff-backup wiki*
- *Automated backups with rdiff-backup*
- *Using rdiff-backup for remote backups*
- *Suppress socket warnings when using rdiff-backup*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux Sample Backup with Duplicity

Duplicity backs directories by producing encrypted tar-format volumes and uploading them to a remote or local file server. Because duplicity uses librsync, the incremental archives are space efficient and only record the parts of files that have changed since the last backup. Because duplicity uses GnuPG to encrypt and/or sign these archives, they will be safe from spying and/or modification by the server.

Installation

Duplicity can be installed in Debian / Ubuntu as simple as:

```
$ sudo apt-get install duplicity ncftp python-boto python-paramiko
```

> *To avoid the **Import of duplicity.backends.giobackend Failed: No module named gio** warning, you need to install the **python-gobject-2** package:*
>
> ```
> $ apt-get install python-gobject-2
> ```

But it not in the CentOS / RedHat default repositories, so you need to install from another source. This script will help in this installation process:

```
#!/bin/bash

wget http://dl.fedoraproject.org/pub/epel/6/x86_64/duplicity-0.6.18-1.el6.x86_64.rpm
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/ncftp-3.2.4-1.el6.x86_64.rpm
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/librsync-0.9.7-15.el6.x86_64.rpm
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/python-GnuPGInterface-0.3.2-6.el6.
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/python-boto-2.5.2-1.el6.noarch.rpm

rpm -Uvh duplicity-0.6.18-1.el6.x86_64.rpm ncftp-3.2.4-1.el6.x86_64.rpm librsync-0.9.
```

Command line basics

To make a backup you can do:

```
$ duplicity --no-encryption /home/openkm file:///path/to/backup
```

To verify it:

```
$ duplicity --no-encryption verify file:///path/to/backup /home/openkm
```

To list backed files:

```
$ duplicity --no-encryption list-current-files file:///path/to/backup
```

To restore a single file:

```
$ duplicity --no-encryption --file-to-restore tomcat-7.0.61/OpenKM.cfg file:///path/to/backup RestoredBackup
```

The path to the file that is to be restored is related to the directory on which the backup set is based. So in the command above, tomcat-7.0.61/OpenKM.cfg plus the directory on which we based our backup (/home/openkm) equals /home/openkm/tomcat-7.0.61/OpenKM.cfg

Preliminaries

In these scripts are assumed these default values:

- ***$TOMCAT_HOME** value is "**/home/openkm/tomcat-7.0.61**"*
- *OpenKM **database** is named "**okmdb**".*
- *OpenKM **database password** value is "***secret***".*
- *OpenKM start & stop **service script** location is "**/etc/init.d/tomcat**"*

Sample script

As a good practice the backup should be done by root user.

The script below does a backup to USB and the application database is MySQL.

USB disk mount point can be defined in /etc/fstab as:

```
/dev/sdb1    /mnt/backup    ext4    defaults    0    0
```

These are the global script sections:

- *Global var configuration under **## BEGIN CONFIG ##** section.*
- *Checking user root under **# Check root** user section.*
- *Deleting older database backup **# Delete older local database backup** section.*
- *Mount disk (optional section) under **# Mount disk** section.*
- *Stop application (optional section) under **# Stop tomcat** section.*
- *Clean logs (optional section) under **# Clean logs** section.*
- *Backup the database to the file system under **#Backup database** section.*
- *Backup repository and database dump to backup destination under **#Backup and purge old backups** section.*
- *Start application (optional section) **# Start tomcat** section.*
- *Show statistics (optional section) **# Status** section.*
- *Un mount disk (optional section) under **# Umount disk** section.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuration parameters explanation:

Parameter	*Description*
HOST	*The server host name.*
DATABASE_PASS	*The database password.* *The data base user to do the backup is always root user.*
OPENKM_DB	*The application database name.* *Usually will be named "okmdb".*
OPENKM_HOME	*The openkm home folder.*
TOMCAT_HOME	*The tomcat home folder.* *Usually into OPENKM_HOME folder.*
DATABASE_EXP	*The database dump folder.* *Usually into the same server.*
BACKUP_DIR	*Backup destination.*

*Create a backup script **/root/backup.sh***

```
$ sudo su
$ vim /root/backup.sh
```

```
#!/bin/bash
#
## BEGIN CONFIG ##
HOST=$(uname -n)
DATABASE_PASS="*secret*"
```

```
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat-7.0.61"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_DIR="file:///mnt/backup"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

# Delete older local database backup
echo -e "### BEGIN: $(date +"%x %X") ###\n"
rm -rf $DATABASE_EXP
mkdir -p $DATABASE_EXP

# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  mount "$BACKUP_DIR";

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "$BACKUP_DIR error mounting";
    exit -1;
  fi
fi

# Stop Tomcat
/etc/init.d/tomcat stop

# Clean logs
#echo "Clean Tomcat temporal files."
#rm -rf $TOMCAT_HOME/logs/*
#rm -rf $TOMCAT_HOME/temp/*
#rm -rf $TOMCAT_HOME/work/Catalina/localhost

# Backup database
if [ -n "$DATABASE_PASS" ]; then
  echo "* Backuping MySQL data from $OPENKM_DB..."
  mysqldump -h localhost -u root -p$DATABASE_PASS $OPENKM_DB > $DATABASE_EXP/mysql_$O
  echo "-------------------------------------";
fi

# Backup and purge old backups
duplicity remove-older-than 1M --force $BACKUP_DIR/$HOST

if [ $(date +%u) -eq 7 ]; then
  echo "*** Full Backup ***"
  duplicity full --no-encryption $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
else
  echo "*** Incremental Backup ***"
  duplicity --no-encryption $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
fi

[ $RETVAL -eq 0 ] && echo "*** SUCCESS ***"
[ $RETVAL -ne 0 ] && echo "*** FAILURE ***"

# Start Tomcat
/etc/init.d/tomcat start
echo -e "\n### END: $(date +"%x %X") ###"

# Status
echo "================================";
```

```
duplicity collection-status $BACKUP_DIR/$HOST
echo "*********************************";
df -h | grep "$BACKUP_DIR"
echo "=================================";

# Umount disk
sync
umount "$BACKUP_DIR"
```

Increase the backup period

*Modify the **# Backup and purge old backups** section increasing the parameter value:*

> *The "**1M**" value indicate will be removed backups older one month. Changing to "**3M**" or "**1Y**" indicate older 3 Month or 1 Year, etc..*

```
--remove-older-than 3M
```

Perform the remote backup

*Modify the **## Config section** section.*

```
BACKUP_DIR="ftp://user@ftp.domain.es/backup"
export FTP_PASSWORD="WhateverPasswordYouSetUp"
```

*Modify the **# Backup and purge old backups** section.*

```
# Backup and purge old backups
duplicity remove-older-than 1M --force $BACKUP_DIR/$HOST

if [ $(date +%u) -eq 7 ]; then
  echo "*** Full Backup ***"
  duplicity full --no-encryption $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
else
  echo "*** Incremental Backup ***"
  duplicity --no-encryption $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
fi

[ $RETVAL -eq 0 ] && echo "*** SUCCESS ***"
[ $RETVAL -ne 0 ] && echo "*** FAILURE ***"
```

*Modify the **# Status** section.*

```
duplicity collection-status $BACKUP_DIR/$HOST
unset FTP_PASSWORD
```

Perform the remote backup to SMB or CIFS

*Modify the **# Mount disk** section.*

```
# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  echo "Mounting $BACKUP_DIR ...";
  mount -t cifs //REMOTE_SERVER/REMOTE_DIR $BACKUP_DIR -o username=REMOTE_USER,passwo

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "Error mounting $BACKUP_DIR.";
    exit -1;
  fi
fi
```

Do the PostgreSQL backup

*Modify the **# Backup database** section*

```
# Backup de PostgreSQL
echo "* Backuping PostgreSQL data from $OPENKM_DB..."
su postgres -c "pg_dump $OPENKM_DB" > $DATABASE_EXP/pg_$OPENKM_DB.sql

# Databases optimizations
su postgres -c "vacuumdb -a -z" > /dev/null
su postgres -c "reindexdb -a -q" 2> /dev/null
```

Do the remote backup to Amazon s3

*Modify the **## Config section** section.*

```
BACKUP_DIR="s3+http://somebucket/somedirectory"
export AWS_ACCESS_KEY_ID=""
export AWS_SECRET_ACCESS_KEY=""
export PASSPHRASE=""
```

*Modify the **# Backup and purge old backups** section.*

```
# Backup and purge old backups
if [ $(date +%u) -eq 7 ]; then
  echo "*** Full Backup ***"
  duplicity full --no-encryption --s3-use-new-style $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
else
  echo "*** Incremental Backup ***"
  duplicity --no-encryption --s3-use-new-style $OPENKM_HOME $BACKUP_DIR/$HOST
  RETVAL=$?
fi

[ $RETVAL -eq 0 ] && echo "*** SUCCESS ***"
```

*Modify the **# Status** section.*

```
# Status
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
duplicity collection-status --s3-use-new-style $BACKUP_DIR/$HOST
unset AWS_ACCESS_KEY_ID
unset AWS_SECRET_ACCESS_KEY
unset PASSPHRASE
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs
$ sudo crontab -e
```

Add these lines according to your personal configuration:

```
MAILTO=nomail@openkm.com
@daily /root/backup.sh | tee /root/logs/backup.$(date +\%Y.\%m.\%d_\%H.\%M.\%S).log
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

Troubleshooting deleting old backups

You have configured Duplicity to make a full backup every 7 days, and deleting backups older than 14 days to save space. What happen if you see a message like this?

```
Tue May  1 00:00:00 2012
Wed May  2 00:00:00 2012
Which can't be deleted because newer sets depend on them
```

Duplicity uses rsync, which contains incremental changes. Those files won't be deleted because, even though the backup maybe older than 7 days, there are backups which are incremental and younger than 7 days.

So, after 2 weeks have passed, those files will be deleted, since the full backup and the incremental backups are now 14 days old, and there exists a full backup newer than the full and incremental backups.

Restoring backup

You can list the available backups:

```
$ duplicity collection-status /path/to/backup
Sun Sep 11 05:00:19 2012       4.25 GB       4.25 GB   (current mirror)
Sun Sep  4 00:00:18 2012       13.3 MB       4.26 GB
Sun Aug 28 00:00:13 2012       674 MB        4.92 GB
Sun Aug 21 00:00:14 2012       5.50 MB       4.93 GB
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
Sun Aug 14 00:00:16 2012      1.75 MB      4.93 GB
Sun Aug  7 00:00:12 2012      288 KB       4.93 GB
Sun Jul 31 00:00:13 2012      43.0 KB      4.93 GB
Fri Jul 29 10:36:39 2012      5.56 KB      4.93 GB
```

Then decide to restore some of the backups, for example the one made on Sun Aug 28 00:00:13 2012:

```
$ duplicity restore --restore-time 2012-08-28 /path/to/backup /path/to/destination
```

*The **--restore-time** parameter accepts several formats. See **duplicity** documentation and Duplicity Backup Howto for more info.*

*Inside of **/path/to/destination** you should see a directory **/home/openkm**, and inside it a couple of directories:*

- ***db**: The backup of the database.*
- ***tomcat-7.0.61**: The backup of the Tomcat installation and OpenKM repository into.*

Additional information

- *Using Duplicity to Backup Your Data*
- *Ubuntu/Debian - Encrypted Incremental Backups With Duplicity on Amazon S3*
- *Duplicity Backup Howto*
- *Duplicity URL Format*
- *Installation and configuration of duplicity for encrypted SFTP remote backup*
- *Playing with Duplicity backup and restore tool and Amazon S3*
- *Bash wrapper script for automated backups with duplicity*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux backup and restore with rsync

Rsync is a software application and network protocol for Unix, Linux and Windows systems which synchronizes files and directories from one location to another while minimizing data transfer using delta encoding when appropriate. An important feature of rsync not found in similar programs/protocols is that the mirroring takes place with only one transmission in each direction. rsync can copy or display directory contents and copy files, optionally using compression and recursion.

Preliminaries

In these scripts are assumed these default values:

- ***$TOMCAT_HOME** value is "**/home/openkm/tomcat-7.0.61**"*
- *OpenKM **database** is named "**okmdb**".*
- *OpenKM **database password** value is "***secret***".*
- *OpenKM start & stop **service script** location is "**/etc/init.d/tomcat**"*

Sample script

As a good practice the backup should be done by root user.

The script below does a backup to USB and the application database is MySQL.

USB disk mount point can be defined in /etc/fstab as:

```
/dev/sdb1    /mnt/backup    ext4    defaults    0    0
```

These are the global script sections:

- *Global var configuration under **## BEGIN CONFIG ##** section.*
- *Checking user root under **# Check root** user section.*
- *Deleting older database dump **# Delete older local database backup** section.*
- *Mount disk (optional section) under **# Mount disk** section.*
- *Stop application (optional section) under **# Stop tomcat** section.*
- *Clean logs (optional section) under **# Clean logs** section.*
- *Backup the database to the filesystem under **#Backup database** section.*
- *Backup repository and database dump to backup destination under **# Create backup** section.*
- *Start application (optional section) **# Start tomcat** section.*
- *Umount disk (optional section) under **# Umount disk** section.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuration parameters explanation:

Parameter	*Description*
HOST	*The server host name.*
DATABASE_PASS	*The database password.* *The database user to do backup is always root user.*
OPENKM_DB	*The application database name.* *Usually will be named "okmdb".*
OPENKM_HOME	*The openkm home folder.*
TOMCAT_HOME	*The tomcat home folder.* *Usually into OPENKM_HOME folder.*
DATABASE_EXP	*The database dump folder.* *Usually into the same server.*
BACKUP_DIR	*Backup destination.*
RSYNC_OPTS	*Rsync parameters.*

*Create a backup script **/root/backup.sh***

```
$ sudo su
$ vim /root/backup.sh
```

```
#!/bin/bash
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
#
## BEGIN CONFIG ##
HOST=$(uname -n)
DATABASE_PASS="*secret*"
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat-7.0.61"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_DIR="/mnt/backup"
RSYNC_OPTS="-apzhR --stats --delete --exclude=*~ --delete-excluded"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

# Delete older local database backup
echo -e "### BEGIN: $(date +"%x %X") ###\n"
rm -rf $DATABASE_EXP
mkdir -p $DATABASE_EXP

# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  mount "$BACKUP_DIR";

  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "$BACKUP_DIR error mounting";
    exit -1;
  fi
fi

# Stop Tomcat
/etc/init.d/tomcat stop

# Clean logs
#echo "Clean Tomcat temporal files."
#rm -rf $TOMCAT_HOME/logs/*
#rm -rf $TOMCAT_HOME/temp/*
#rm -rf $TOMCAT_HOME/work/Catalina/localhost

# Backup database
if [ -n "$DATABASE_PASS" ]; then
  echo "* Backuping MySQL data from $OPENKM_DB..."
  mysqldump -h localhost -u root -p$DATABASE_PASS $OPENKM_DB > $DATABASE_EXP/mysql_$O
  echo "-------------------------------------";
fi

# Create backup
rsync $RSYNC_OPTS $OPENKM_HOME $BACKUP_DIR/$HOST

# Start Tomcat
/etc/init.d/tomcat start
echo -e "\n### END: $(date +"%x %X") ###"

# Umount disk
sync
umount "$BACKUP_DIR"
```

Do backup with rotation

Modify the ***# Create backup*** *section.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
# Create backup
# Calculate snapshot
LAST_SNAPSHOT=`ls -ltr $BACKUP_DIR | tail -1 | awk {'print $8'} | cut -d . -f 2`
NEW_SNAPSHOT=$((LAST_SNAPSHOT+1))

rsync $RSYNC_OPTS --link-dest="$BACKUP_DIR/$HOST/backup.$LAST_SNAPSHOT" $FILES "$BACK
```

Other configuration:

```
# Create backup
"cd $BACKUP_DIR/$HOST; rm -rf backup.3; mv backup.2 backup.3; mv backup.1 bac
rsync $RSYNC_OPTS --link-dest="$BACKUP_DIR/$HOST/backup.1" "OPENKM_HOME" "$BA
```

Do remote backup

*Modify the **## BEGIN CONFIG ##** section.*

```
BACKUP_SRV="user@server.com"
```

*Modify the **# Create backup** section.*

```
# Create backup
# Create missing backup folder ( optional )
ssh $BACKUP_SRV "mkdir -p $BACKUP_DIR"

# Calculate snapshot
NEW_SNAPSHOT=$(date "+%Y-%m-%dT%H:%M:%S")
LAST_SNAPSHOT=$(ssh $BACKUP_SRV ls -tr $BACKUP_DIR/$HOST | tail -1)

if [ $LAST_SNAPSHOT ]; then
  echo "Incremental backup $NEW_SNAPSHOT based on $LAST_SNAPSHOT... "
  rsync $RSYNC_OPTS --link-dest="$BACKUP_DIR/$HOST/$LAST_SNAPSHOT" "$OPENKM_HOME" "$B
else
  echo "Initial full backup $NEW_SNAPSHOT..."
  rsync $RSYNC_OPTS "$OPENKM_HOME" "$BACKUP_SRV:$BACKUP_DIR/$HOST/$NEW_SNAPSHOT"
fi
```

More simple configuration:

```
rsync $RSYNC_OPTS "$OPENKM_HOME" "$BACKUP_SRV:$BACKUP_DIR/$HOST"
```

Do incremental backup with backup-dir

*Modify the **## BEGIN CONFIG ##** section.*

```
RSYNC_OPTS="-ahR --partial --stats --delete --exclude=*~ --exclude=tomcat-7.0.61/temp
```

*Modify the **# Create backup** section.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
# Create backup
NEW_SNAPSHOT=`date +%Y-%m-%d`
echo -e "\n### RSYNC - BEGIN : $(date +"%x %X") ###\n"
echo "* New Snapshot: $NEW_SNAPSHOT"
rsync $RSYNC_OPTS --backup --backup-dir="$BACKUP_DIR/$HOST/INCREMENTAL/$NEW_SNAPSHOT"
```

Do incremental backup with link-dest

Modify the ***## BEGIN CONFIG ##*** *section.*

```
RSYNC_OPTS="-ahR --partial --stats --delete --exclude=*~ --exclude=tomcat-7.0.61/temp
```

Modify the ***# Create backup*** *section.*

```
# Create backup
# Calculate snapshot
LAST_SNAPSHOT=`ssh $BACKUP_SRV "cd $BACKUP_DIR && stat --format='%n' * | tail -1"`
NEW_SNAPSHOT=`date +%Y-%m-%d`
echo -e "\n### RSYNC - BEGIN : $(date +"%x %X") ###\n"
echo "* Last Snapshot: $LAST_SNAPSHOT"
echo "* New Snapshot: $NEW_SNAPSHOT"

if [ ! $LAST_SNAPSHOT ]; then
  rsync $RSYNC_OPTS $OPENKM_HOME/ "$BACKUP_DIR/$HOST/$NEW_SNAPSHOT"
else
  rsync $RSYNC_OPTS --link-dest="$BACKUP_DIR/$HOST/$LAST_SNAPSHOT" $OPENKM_HOME/ "$BA
fi
```

Do remote backup to SMB or CIFS

Modify the ***# Mount disk*** *section.*

```
# Mount disk
if mount | grep "$BACKUP_DIR type" > /dev/null; then
  echo "$BACKUP_DIR already mounted";
else
  echo "Mounting $BACKUP_DIR ...";
  mount -t cifs //REMOTE_SERVER/REMOTE_DIR $BACKUP_DIR -o username=REMOTE_USER,passwo
  if mount | grep "$BACKUP_DIR type" > /dev/null; then
    echo "$BACKUP_DIR mounted";
  else
    echo "Error mounting $BACKUP_DIR.";
    exit -1;
  fi
fi
```

Do PostgreSQL backup

Modify the ***# Backup database*** *section*

```
# Backup de PostgreSQL
echo "* Backuping PostgreSQL data from $OPENKM_DB..."
```

```
su postgres -c "pg_dump $OPENKM_DB" > $DATABASE_EXP/pg_$OPENKM_DB.sql

# Databases optimizations
su postgres -c "vacuumdb -a -z" > /dev/null
su postgres -c "reindexdb -a -q" 2> /dev/null
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs

$ sudo crontab -e
```

And add these lines according to your personal configuration:

```
MAILTO=nomail@openkm.com
@daily /root/backup.sh | tee /root/logs/backup.$(date +\%Y.\%m.\%d_\%H.\%M.\%S).log
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

Rsync and FTP

*Rsync need SSH access to work, but you can also use it with an FTP server if you mount it locally with **curlftpfs**:*

Backup performance may degrade using this method.

Install curlftpfs and rsync:

```
$ sudo apt-get install curlftpfs rsync
```

Create mountpoin:

```
$ mkdir /path/to/mount
```

Mount remote director:

```
$ curlftpfs ftp.example.com /path/to/mount
```

Create local directory:

```
$ mkdir /local/path
```

Rsync mounted directory to local director:

```
$ rsync -r -t -v --progress --bwlimit=500 /path/to/mount/ /local/path/
```

Restoring backup

You can list the available backups:

```
$ ls -ld /path/to/backup/*
drwxr-xr-x 4 root root 4096 sep 11 05:00:19 2012 2012-09-11T05:00:19
drwxr-xr-x 4 root root 4096 sep  4 00:00:18 2012 2012-09-04T00:00:18
drwxr-xr-x 4 root root 4096 aug 28 00:00:13 2012 2012-08-28T00:00:13
drwxr-xr-x 4 root root 4096 aug 21 00:00:14 2012 2012-08-21T00:00:14
drwxr-xr-x 4 root root 4096 aug 14 00:00:16 2012 2012-08-14T00:00:16
drwxr-xr-x 4 root root 4096 aug  7 00:00:12 2012 2012-08-07T00:00:12
drwxr-xr-x 4 root root 4096 jul 31 00:00:13 2012 2012-07-31T00:00:13
drwxr-xr-x 4 root root 4096 jul 29 10:36:39 2012 2012-07-29T10:36:39
```

Then decide to restore some of the backups, for example the one made on Sun Aug 28 00:00:13 2012:

```
$ cp /path/to/backup/2012-08-28T00:00:13 /path/to/destination
```

*See **rsync** documentation for more info.*

*Inside of **/path/to/destination** you should see a directory **/home/openkm**, and inside it a couple of directories:*

- ***db**: The backup of the database.*
- ***tomcat-7.0.61**: The backup of the Tomcat installation and OpenKM repository into.*

Additional information

- *Easy Automated Snapshot-Style Backups with Linux and Rsync*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Linux backup with LFTP

LFTP is a sophisticated FTP client, and a file transfer program supporting a number of network protocols.

This configuration does not provide incremental backups.

Preliminaries

In these scripts are assumed these default values:

- ***$TOMCAT_HOME** value is "**/home/openkm/tomcat-7.0.61**"*
- *OpenKM **database** is named "**okmdb**".*
- *OpenKM **database password** value is "***secret***".*
- *OpenKM start & stop **service script** location is "**/etc/init.d/tomcat**"*

Sample script

As a good practice the backup should be done by root user.

The script below does a backup to USB and the application database is MySQL.

USB disk mount point can be defined in /etc/fstab as:

```
/dev/sdb1    /mnt/backup    ext4    defaults    0    0
```

These are the global script sections:

- *Global var configuration under **## BEGIN CONFIG ##** section.*
- *Checking user root under **# Check root** user section.*
- *Deleting older database backup **# Delete older local database backup** section.*
- *Stop application (optional section) under **# Stop tomcat** section.*
- *Clean logs (optional section) under **# Clean logs** section.*
- *Backup the database to the filesystem under **#Backup database** section.*
- *Backup repository and database dump to backup destination under **#Backup and purge old backups** section.*
- *Start application (optional section) **# Start tomcat** section.*

Configuration parameters explanation:

Parameter	*Description*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

HOST	*The server host name.*
NOW	*Actual date in format 'dd-MM-yyyy'.*
DATABASE_PASS	*The database password.* *The data base user to do backup is always root user.*
OPENKM_DB	*The application database name.* *Usually will be named "okmdb".*
OPENKM_HOME	*The openkm home folder.*
TOMCAT_HOME	*The tomcat home folder.* *Usually into OPENKM_HOME folder.*
DATABASE_EXP	*The database dump folder.* *Usually into the same server.*
BACKUP_DIR	*FTP Backup destination.*

*Create a backup script **/root/backup.sh***

```
sudo su

vim /root/backup.sh
```

```
#!/bin/bash
#
## BEGIN CONFIG ##
HOST=$(uname -n)
NOW=$(date +"%d-%m-%Y")
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
DATABASE_PASS="*secret*"
OPENKM_DB="okmdb"
OPENKM_HOME="/home/openkm"
TOMCAT_HOME="$OPENKM_HOME/tomcat-7.0.61"
DATABASE_EXP="$OPENKM_HOME/db"
BACKUP_DIR="ftp://user:password@ftp.server/backup"
## END CONFIG ##

# Check root user
if [ $(id -u) != 0 ]; then echo "You should run this script as root"; exit; fi

# Delete older local database backup
echo -e "### BEGIN: $(date +"%x %X") ###\n"
rm -rf $DATABASE_EXP
mkdir -p $DATABASE_EXP

# Stop Tomcat
/etc/init.d/tomcat stop

# Clean logs
#echo "Clean Tomcat temporal files."
#rm -rf $TOMCAT_HOME/logs/*
#rm -rf $TOMCAT_HOME/temp/*
#rm -rf $TOMCAT_HOME/work/Catalina/localhost

# Backup database
if [ -n "$DATABASE_PASS" ]; then
  echo "* Backuping MySQL data from $OPENKM_DB..."
  mysqldump -h localhost -u root -p$DATABASE_PASS $OPENKM_DB > $DATABASE_EXP/mysql_$O
  echo "-------------------------------------";
fi

# Backup and purge old backups
lftp $BACKUP_DIR -e "mkdir $NOW; mirror --reverse $OPENKM_HOME $NOW; exit"

# Start Tomcat
/etc/init.d/tomcat start
echo -e "\n### END: $(date +"%x %X") ###"
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs

$ sudo crontab -e
```

Add these lines according to your personal configuration:

```
MAILTO=nomail@openkm.com
@daily /root/backup.sh | tee /root/logs/backup.$(date +\%Y.\%m.\%d_\%H.\%M.\%S).log
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

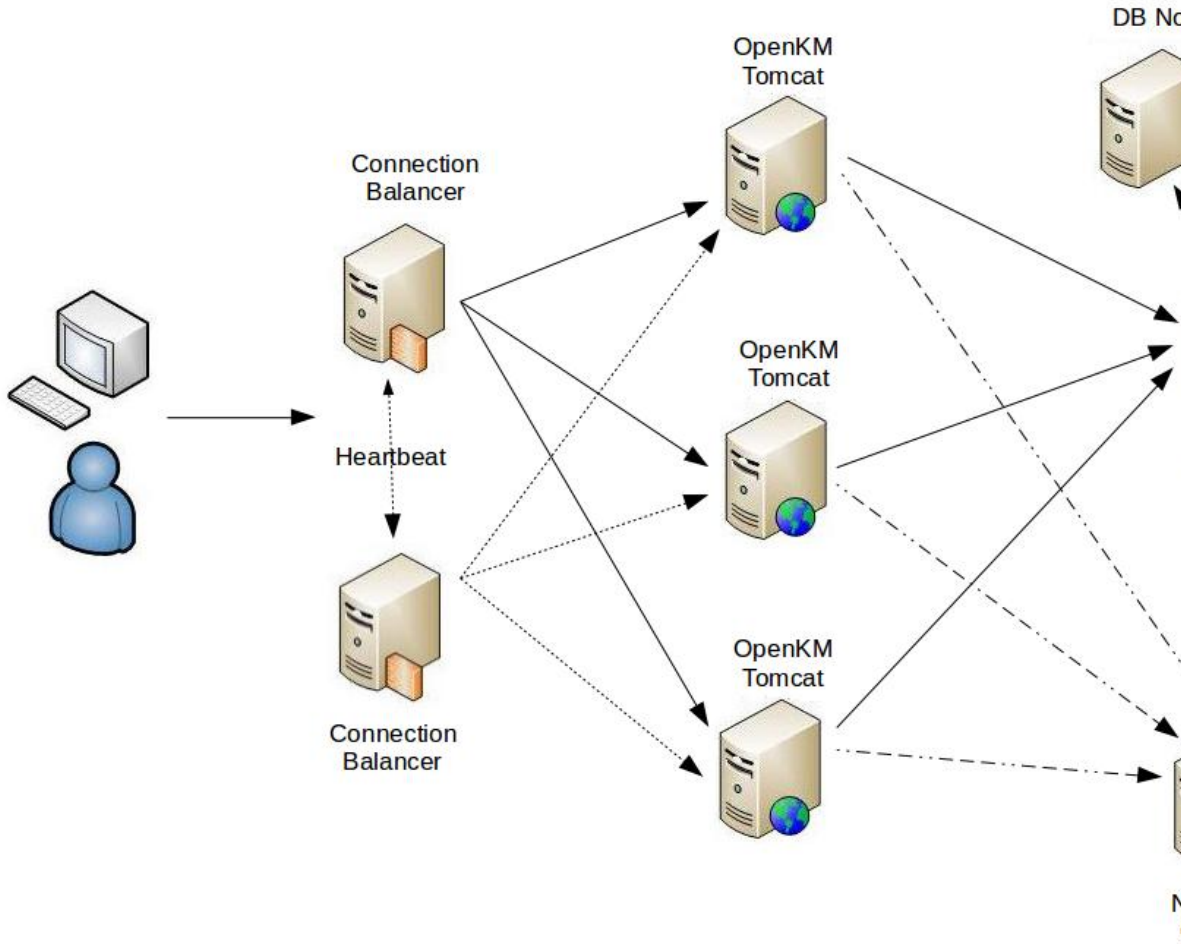
Cluster configuration

This new cluster arquitecture is an enhancement of the previous one, and improves these problems found in the other version:

- ***Single error point**: Only master can modify the index. If the master fails the whole system would have problems.*
- ***Latency**: Slave nodes have to refresh their local index every some minutes. So there is a latency between master and slave to see a newly indexed item.*

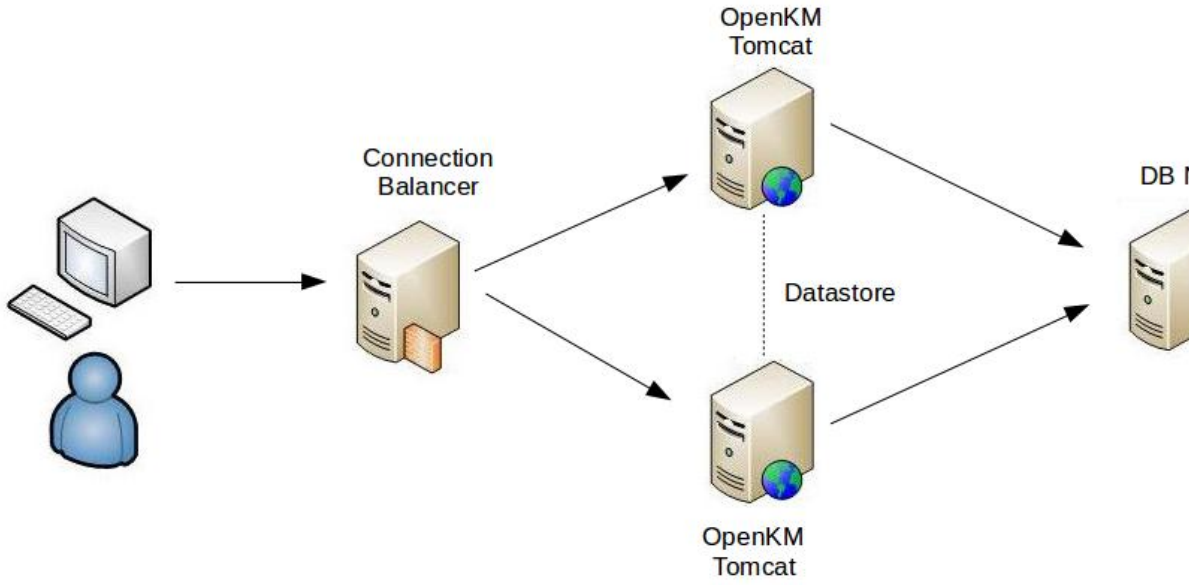
This kind of aquitecture provides server load spreading as well as high availability. Let's take a look at the proposed architecture. There are several OpenKM server and one (or several) HAProxy server which will distribute the user petitions to these OpenKM servers. HAProxy can be also configured to detect which OpenKM server is down and avoid sending petitions until it is up again. Optionally you can configure several replicated database servers (also handled by another HAProxy) which prevent the service from being interrupted in case of a database node goes down. By default OpenKM stores the documents in disk to maximise performance, so you need these files replicated across your OpenKM servers. An option is GlusterFS which is a software solution to create distributed filesystems, but NFS is quite good in most situations and easier to setup in case of Linux servers.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Keep on mind that this is only a proposal and you are free to implement whatever infrastructure you want to achieve the same results. In order to avoid an excesive and difficult configuration, we are going to describe a simplified architecture which will be provide the same objective: the master node, the database and the NFS server are in the same machine.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*Remember that every OpenKM instance should use the same database instance. Only one database for all the instances. This resource is configured in **TOMCAT_HOME/openkm.properties** file.*

If you look into the TOMCAT_HOME/repository directory you can see these others:

- ***cache**: These are generated files used mostly in preview. In case of being deleted they will be re-created again.*
- ***datastore**: There files are the binary content of the documents stored in OpenKM. You should take care of them, and a backup is always recommended.*
- ***index**: Where Lucene stores the indexes used in document and metadata search. In case of deletion you can re-index the repository from Administration > Utilities.*

*According to this information, you can place **cache** and **datastore** folders in another server as shared resources. In average, 80% of user actions are read documents and only 20% write (create or update documents). The problematic one is the **index** folder because an Lucene indexes can't be shared between several OpenKM instances. So, what can we do? The solution is that each OpenKM instance has his own index and they send messages between them to keep their local index updated. If a node need to update the index, it also send a the message to the other nodes which will process the modification locally.*

Proxy node

This is the only node which should be accesible by the users and will dispatch the petitions between the slave nodes. The most important part is the HAProxy configuration. This configuration format works on HAProxy v1.5

```
frontend LB
```

```
  bind 192.168.0.220:80
  reqadd X-Forwarded-Proto:\ http
  default_backend LB

 backend LB 192.168.0.220:80
  mode http
  stats enable
  stats hide-version
  stats uri /stats
  stats realm HAProxy\ Statistics
  stats auth admin:admin
  balance roundrobin
  option httpchk
  option httpclose
  option forwardfor
  cookie LB insert
  server node1 192.168.0.222:8080 cookie node1 check
  server node2 192.168.0.223:8080 cookie node2 check
```

Accooding to this configuration, user petitions will be balanced using the round-robin algorithm between the two OpenKM servers. In case of any of the OpenKM server fails the petitions will be only forwarded to the working one. At http://proxy/stats (default user: admin, password: admin) you will be able to see the nodes status and other statistics. A complete manual of HAProxy is available at http://cbonte.github.io/haproxy-dconv/configuration-1.5.html. If you use another HAProxy version some options may have changed.

Master node

This kind of node is not very different from the slaves. The only real difference is it can run crontab task among other administration actions. As we said previously to simplify the configuration this master node will also store the ***cache*** *and* ***datastore*** *files.*

> *In this example, we are going to use this server also as NFS server, but it's very common to have a SAN which acts as the NFS server.*

Let's create these directories:

```
$ sudo mkdir /mnt/okm_cache
$ sudo mkdir /mnt/okm_dstore
$ sudo mkdir /mnt/okm_extraction
$ sudo chown openkm:openkm /mnt/okm_*
```

We are expose these directories by NFS so we need to install support:

```
$ sudo apt-get install nfs-kernel-server
```

Once installed, edit the ***/etc/exports*** *file and add these lines:*

```
/mnt/okm_cache *(rw,sync,no_root_squash,no_subtree_check)
/mnt/okm_dstore *(rw,sync,no_root_squash,no_subtree_check)
```

```
/mnt/okm_extraction *(rw,sync,no_root_squash,no_subtree_check)
```

After saving it, you have to execute this command (every time you modify this file you should execute it):

```
$ sudo exportfs -ra
```

Now let's make the symbolic links:

```
$ sudo ln -s /mnt/okm_cache TOMCAT_HOME/repository/cache
$ sudo ln -s /mnt/okm_dstore TOMCAT_HOME/repository/datastore
$ sudo ln -s /mnt/okm_extraction TOMCAT_HOME/repository/extraction
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/cache
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/datastore
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/extraction
```

Instead of making links you can configure OpenKM to create these directories into other locations:

```
repository.extraction.home=/mnt/okm_extraction
repository.datastore.home=/mnt/okm_dstore
repository.cache.home=/mnt/okm_cache
```

*To enable this configuration edit the **TOMCAT_HOME/openkm.properties** and uncomment this part:*

```
# Cluster - Master
cluster.enabled=true
cluster.node=master
```

Slave nodes

*These nodes can write to **cache** and **datastore** folders. Here we also need to create the shared directories which be used as mount point:*

```
$ sudo mkdir /mnt/okm_cache
$ sudo mkdir /mnt/okm_dstore
$ sudo mkdir /mnt/okm_extraction
$ sudo chown openkm:openkm /mnt/okm_*
```

Install NFS client support:

```
$ sudo apt-get install nfs-common
```

*We want them to be mounted automatically, so edit the **/etc/fstab** file and append:*

```
# Cluster
```

```
master:/mnt/okm_cache /mnt/okm_cache nfs auto,noatime,rsize=8192,wsize=8192,timeo=14,
master:/mnt/okm_dstore /mnt/okm_dstore nfs auto,noatime,rsize=8192,wsize=8192,timeo=1
master:/mnt/okm_extraction /mnt/okm_extraction nfs auto,noatime,rsize=8192,wsize=8192
```

Edit the /etc/hosts file and configure it to resolve the "master" server name to the right IP.

Now let's mount them:

```
$ sudo mount -a
```

Let's make the symbolic links:

```
$ sudo ln -s /mnt/okm_cache TOMCAT_HOME/repository/cache
$ sudo ln -s /mnt/okm_dstore TOMCAT_HOME/repository/datastore
$ sudo ln -s /mnt/okm_extraction TOMCAT_HOME/repository/extraction
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/cache
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/datastore
$ sudo chown -h openkm:openkm TOMCAT_HOME/repository/extraction
```

Instead of making links you can configure OpenKM to create these directories into other locations:

```
repository.extraction.home=/mnt/okm_extraction
repository.datastore.home=/mnt/okm_dstore
repository.cache.home=/mnt/okm_cache
```

*To enable this configuration edit the **TOMCAT_HOME/openkm.properties** and uncomment this part:*

```
# Cluster - Slave
cluster.enabled=true
cluster.node=slave
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Sharding configuration

With sharding, OpenKM can distribute the repository files into different locations. This feature can help in case the main disk is full, so new documents will be placed in a another disk.

The best way to understand sharing it is to think that each shard is a different disk.

All documents will be stored in the main shard, until a new shard is created and set as current shard. Shard can be managed from administration and these options are available:

- *Create a new shard*
- *Set a shard as current*

Deleting a shard is not possible because is dangerous and may cause data destruction.

A use case of sharding is to have a fast media like SSD (which is also expensive) in the main shard, and other slower and cheaper HDD in the second shard. Documents will be moved from the fast main shard to the second slow one by a process which can identify less accessed documents. This way the fast shard will be filled with frequently used documents, which also will improve the overall system performance.

By default, there is only one shard and a specific configuration is needed to make use of this new feature.

In order to configure sharding, use this configuration property:

```
repository.datastore.home=${repository.home}/${shard.id}/${tenant.id}/datastore
```

The key is the "shard.id" variable which will be replaced in OpenKM by the currently enabled shard. So, if you are in shard #1 and tenant #2, the datastore location will be ${repository.home}/shard_1/tenant_1/datastore.

Currently only the repository can be divided in shards.

*The **repository.datastore.home** configuration property should be set before uploading any documents to OpenKM otherwise the documents binaries won't be accessible unless you move them to the new location.*

Sharding and Docker

*In case you are using Docker to deploy OpenKM we recommend to externalize the **openkm.properties** configuration property file to be able to configure sharding.*

This is a sample Docker Compose definition:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
version: '3.2'

services:

  openkm_mysql:
    image: docker.openkm.com/private/professional:7.1.14
    container_name: openkm_mysql_shard
    hostname: openkm
    ports:
      - 8080:8080
    volumes:
      - ${PWD}/openkm.properties:/opt/tomcat/openkm.properties
      - ${PWD}/repository:/opt/tomcat/repository
    environment:
      - TZ=Europe/Madrid
    entrypoint: [ "wait-for-it.sh", "mysql:3306", "--timeout=0", "--strict", "--", "e
    links:
      - mysql:mysql
    networks:
      skynet:
        ipv4_address: 172.28.1.1

  mysql:
    image: mysql:8.0.18
    container_name: okmdb_mysql_shard
    hostname: mysql
    command: --default-authentication-plugin=mysql_native_password --character-set-se
    volumes:
      - ${PWD}/mysql:/var/lib/mysql
    environment:
      - MYSQL_DATABASE=okmdb
      - MYSQL_USER=openkm
      - MYSQL_PASSWORD=openkm
      - MYSQL_ROOT_PASSWORD=openkm
    security_opt:
      - seccomp:unconfined
    networks:
      skynet:
        ipv4_address: 172.28.1.2

volumes:
  openkm_mysql_shard:
  okmdb_mysql_shard:

networks:
  skynet:
    ipam:
      driver: default
      config:
        - subnet: 172.28.0.0/16
```

*And this is the according **openkm.properties** (shards with no tenants):*

```
# OpenKM Hibernate configuration values
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://mysql:3306/okmdb?useUnicode=true&characterEncoding
spring.datasource.username=openkm
spring.datasource.password=openkm

# JPA stuff
spring.jpa.hibernate.ddl-auto=create-only
spring.jpa.properties.hibernate.dialect=com.openkm.db.dialect.MySQL5InnoDBDialect

# Logging config
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

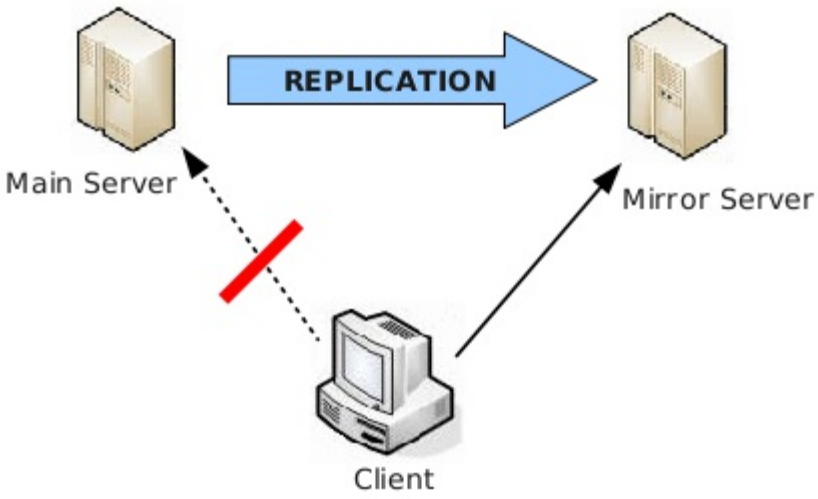
```
spring.output.ansi.enabled=always

# Extra
repository.datastore.home=/opt/tomcat/repository/${shard.id}/datastore
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Instance replication

To enhance OpenKM availability you can have two instances of the application running on different servers. If the principal server gets down due to a hardware failure you can switch to the mirrored server and keep working.

If you want a typical high availability configuration you can use the [*Cluster configuration*] *but it is a complex architecture if you want to achieve a real high availability scheme. There is another option, easier to setup and also provides a good protection against a server failure. This is somewhat like a simplified cluster configuration where the main instance is synchronized with the backup instance. For example, this can be done using a software solution like Rsync what minimizes the transfer data.*



In this scheme, we assume:

- *Each node has its own document repository and database.*
- *The backup instance is supposed to be a clone of the main instance.*
- *In case the main instance is down, you should send the client request to the backup instance.*

An option is to use a DNS which you can modify in case of server failure, or an HAProxy instance which properly configured can detect a problem in the main server and forward the request to the working one.

Keep in mind that in this configuration only one server should accept client petitions.

When the main server is down, the backup one becomes the new main server so the old main server is now the backup one (well, when it is available again).

This is the configuration of each server:

Main server

- *OpenKM is configured normally.*
- *The database is configured to send modifications to the backup instance.*
- *Rsync is configured to send datastore modifications every few minutes to the backup server.*

Mirror server

- *OpenKM is configured in read-only mode.*
- *The database is configured to accept modification from the main instance.*
- *Datastore files are periodically updated by Rsync tool from the main server.*

Different implementations

The provided solution is not the only one, but the simplest.

An improvement over this first approach is using an external file server (connected by NFS) so Rsync is not needed. Using a NAS solution which improves the data backup and availability. It is highly recommended if the stored information is critical for your business.

Another improvement is related to the database. Some database can be configured in master / slave mode, which the required mode for this server configuration. But you can also deploy the database in a dedicated server (which should be also replied) but some providers like Oracle offer specific database cluster solutions which should be taken in consideration.

Sample Linux implementation

The replication is done only in one direction, from main server to mirror server.

The mirror server must be configured to be in read-only mode so users can't add or modify documents.

The follow script will propagate the repository changes from main server to mirror server:

On the sample the main server 192.168.1.101.

Both OpenKM are installed at the OpenKM user home (/home/openkm) for simplicity.

This script should be executed - by user root - from the main server, and can be scheduled to be executed every day using Linux cron utility, for example.

The script uses rsync to minimize network load and only modified or added documents will be transferred. The whole process can take a few minutes, depending on your repository activity.

```
#!/bin/bash
TOMCAT_HOME="/home/openkm/tomcat-8.5.24"
REMOTE_HOST="root@192.168.1.102"
MYSQL_PASS="*****"

echo -e "### BEGIN: $(date +"%x %X") ###\n"
```

```
# Stop local OpenKM
/etc/init.d/openkm stop

# Stop remote OpenKM
ssh $REMOTE_HOST '/etc/init.d/openkm stop'

# Sync OpenKM repositories
rsync -avh --stats --delete --exclude repository/.system.key --exclude logs $TOMCAT_H

# Dump and copy database
mysqldump -u root -p${MYSQL_PASS} okmdb | ssh $REMOTE_HOST "mysql -u root -p${MYSQL_P

# Start local OpenKM
/etc/init.d/openkm start

# Start remote OpenKM
ssh $REMOTE_HOST '/etc/init.d/openkm start'

echo -e "\n### END: $(date +"%x %X") ###"
```

Copy the following SQL into the remote $TOMCAT_HOME/start.sql server localtion.

```
update OKM_CONFIG set 'true' where CFG_KEY='system.readonly';
```

After the first synchronization, stop remote Tomcat, update the start.sql script there and start the Tomcat again.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

How to move to another server

Although the system should not be affected, before doing this kind of action we recommend having a backup. Refer to the Backup and restore section.

Linux

In the next documentation, will refer to the Installing on Ubuntu and Debian in some sections because are the same actions.

Preliminaries

- *Ensure you **have enough disk space** in the new server.*
- *Check if you have **enough CPU power and RAM** in the new server.*
- ***Create the user** who's the **owner** of the OpenKM files ($TOMCAT_HOME folder). This user is usually named "**openkm**" (more info at Installing on Ubuntu and Debian), but maybe in the installation was choose other one.*
- *Update the OS.*

Stop OpenKM service

It is important while moving the data between the two servers do not perform any kind of changes in the old server. That's why is a good practice to have OpenKM stopped.

*If you **do not want to stop the OpenKM service** , set OpenKM as read-only with a configuration parameter named "**system.readonly**".*

Backup the database

*In the "**server.xml**" file (at $TOMCAT_HOME/conf folder) is configured the database connectio. In the next XML sample the user is named "**openkm**" with password "**openkm**" and the connection URL is "**jdbc:mysql://localhost:3306/okmdb?autoReconnect=true&useUnicode=true&characterEncoding=UTF8**":*

```
<Resource name="jdbc/OpenKMDS" auth="Container" type="javax.sql.DataSource"
          maxActive="100" maxIdle="30" maxWait="10000" validationQuery="select 1"
          username="openkm" password="openkm" driverClassName="com.mysql.jdbc.Driver"
          url="jdbc:mysql://localhost:3306/okmdb?autoReconnect=true&amp;useUnicode=tr
```

*The command below explains how to backup the MySQL and MariaDB database. The database is usually named "**okmdb**":*

```
$ mysqldump -h localhost -u openkm -popenkm okmdb > okmdb.sql
```

Restore the database

- ***Install the same database engine*** *in the new server.*
- ***Create the database user with the same credentials*** *(more info at Installing on Ubuntu and Debian).*
- ***Copy the okmdb.sql file to the new server*****.**
- ***Restore the backup*****.**

The sample below explains how to restore the MySQL and MariaDB database. Again, the database is usually named " ***okmdb****":*

```
$ mysql -h localhost -u openkm -popenkm okmdb < okmdb.sql
```

Copy binary files

Take a look at the file OpenKM.cfg, the whole repository is usually in the TOMCAT_HOME folder, but depending on your configuration it may be has configured in other location. This repository folder should also be copied.

Take a look at the parameters in Change repository home.

- ***Copy the TOMCAT_HOME folder to the new server*****.** *We suggest to* ***keep the same origin folder path in destination*** *(otherwise you would need to make extra configuration steps).*
- *Set* ***grants to the owner of the TOMCAT_HOME folder in the new server*** *(usually the user named openkm).*

The sample below shows how to copy it (from the old server to the new server):

```
$ rsync -avh --progress --stats /home/openkm/tomcat-8.5.57 root@newServer:/home/openkm/tomcat-8.5.57
```

The sample below shows how to set the owner (you need to execute as root):

```
$ chown openkm:openkm -Rf /home/openkm/tomcat-8.5.57
```

Configure service in the new server

- *Copy the file* ***/etc/init.d/openkm to the new server at the same location and register as a service*** *. (more info at Installing on Ubuntu and Debian).*

Install third-party tools

In OpenKM Administration > Configuration parameters you will find information about most applications required. Some of them are under $TOMCAT_HOME/bin folder, so these have been just copied in previous steps as part of the TOMCAT_HOME folder.

The minimum required packages are (more info at Installing on Ubuntu and Debian):

- *ImageMagick*
- *Ghostscript*
- *LibreOffice*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Tesseract OCR engine (remember the language packs required in your environment)*

Start OpenKM service

Until the license is not moved to the new server you will only be able to access the administration user interface.

- ***In case the new server has less RAM than the old server you need to check*** *the* ***$TOMCAT_HOME/bin/setenv.sh*** *file and adapt properly.*
- *Check the openkm.log file for the startup process and look for possible errors.*

Additional steps

- *If the new server has a different number of cores you need to check the value of the configuration parameters named "****text.extraction.threads****" and "****text.extraction.batch****".*
- ***Open administration*** *at http://localhost:8080/openkm/management/ and* ***execute a repository checker*** *(this action may take a lot of time until completed).*

Moving license

Coordinate with the OpenKM team to move the license from the old server to the new one.

Final checks

- *Check the preview of different document types (PDF, Microsoft Office).*
- *Check the thumbnails of different document types (PDF, Microsoft Office).*
- *Create a folder.*
- *Upload a new document.*
- *Download a previously existing document.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Setup

*This section provides information for **configuring** and **tuning** OpenKM.*

The application contains a lot of Configuration parameters and probably you will not need to know all of them. However, we encourage you to take a look at the Recommended configuration parameters section where you will find the minimum configuration required to use OpenKM.

Also you could be interested on:

- *Preview parameters*
- *User interface configuration parameters*

OpenKM works with third-party applications Wordpress, Joomla or Google Drive among others that need specific setup to get it running.

If you have some issue see Troubleshooting section.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Antivirus configuration

OpenKM can be integrated with an anti virus software which is be able to execute anti virus check from command line.

The example below is based on ClamAV, an Open Source anti virus engine which detects trojans, viruses, malware & other malicious programs.

Linux

Debian / Ubuntu

Install antivirus:

```
$ sudo aptitude install clamav
```

*Go to **Administration > Configuration parameters**:*

Field / Property	Type	Description
system.antivir	**String**	/usr/bin/clamscan
background.antivirus.check	**Boolean**	true
background.antivirus.check.threads	**Integer**	20
background.antivirus.check.batch	**Integer**	2

More information about configuration parameters:

- *Antivirus configuration parameters*
- *Performance configuration parameters*

Red Hat / CentOS

ClamAV is not present by default in default repositories, so you must install it from a non default one.

Repository valid for Centos 5.2 (for other versions it may be different):

```
[dag]
name=Dag RPM Repository for Red Hat Enterprise Linux
baseurl=http://apt.sw.be/redhat/el$releasever/en/$basearch/dag/
gpgcheck=1
gpgkey=http://dag.wieers.com/packages/RPM-GPG-KEY.dag.txt
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
enabled=1
```

Install antivirus:

```
$ yum install clamd
```

Start the daemon:

```
$ /etc/init.d/clamd start
```

Update the virus database:

```
$ freshclam
```

Windows

To Install and configure ClamAV on Windows:

Note: ***GREEN Text*** *like this denotes commands run in a command console/terminal*

BLACK Text *like this denotes code changes*

- *Download the newest version from: https://www.clamav.net/downloads*
- *Unpack the ZIP archive and move it to: C:\ so that it appears in your path as: C:\ClamAV*
- *Copy the example conf files from C:\ClamAV\conf_examples to C:\ClamAV*
- *Open up a command terminal: Start->Run* ***cmd***
- *Change to the ClamAV directory:* ***cd C:\ClamAV***
- *Run* ***clamconf.exe****, it should inform that you need to edit the example config file. (I included this just so you can check that the files are in the right location)*

For DEFAULT MINIMAL configuration (Probably not suitable and you should configure the parameters for your situation) Edit freshclam.conf, clamav-milter.conf and comment ***Example****. In addition to commenting* ***Example*** *in clamd.conf; to enable the local server you will need to change line 80 to:* ***LocalSocket C:\CLAMAV\clamd.socket***

- *Run* ***clamconf.exe*** *again, you will get an output of all your configurations.*

At the beginning you will get a message about being unable to open database, this is because it has not yet been created.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Run **`mkdir database`**. If you run clamconf again you will see that the database error is gone and that you have 0 signatures*
- *Run **`freshclam.exe`**, this will start the initial signature download for ClamAV and may take a few minutes.*

*At this point you can run **`clamd.exe`** to check that the server can run manually. (If you decide to test run at this point, you will have to kill the process for the next section)*

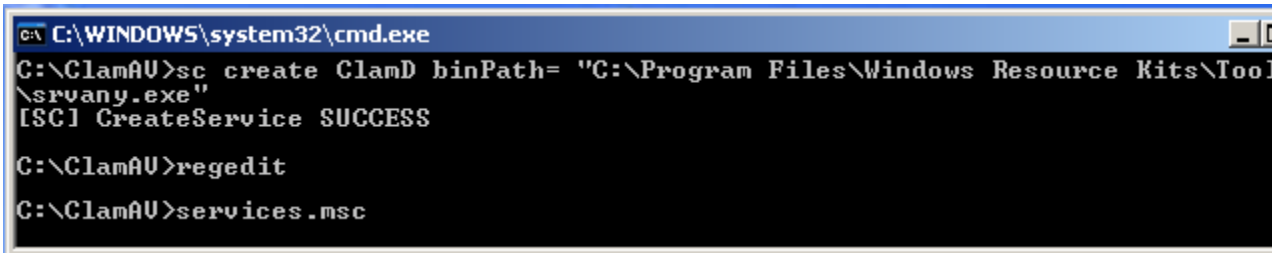
In case you need to create a freshclam.conf file use this as base:

```
# URL of server where database updates are to be downloaded from
# If this option is given multiple times, each will be tried in
# the order given until an update is successfully downloaded
#MAKE SYU
DatabaseDirectory "C:\Program Files\ClamAV-x64\db\"
DatabaseMirror database.clamav.net
# Number of times to try each mirror before moving to the next one
MaxAttempts 3
#LogFile C:\Program Files\ClamAV-x64\log\freshclam.log
#NotifyClamd C:\Program Files\ClamAV-x64\clamd.conf
LogFileMaxSize 20480000
LogTime true
UpdateLogFile C:\Program Files\ClamAV-x64\log\freshclam.log
```

Clamd Service Setup

Now to setup it up so it starts automatically:

Details for the screen capture are below:



- *Run **`sc create ClamD binPath= C:\Program Files\Windows Resource Kits\Tools\srvany.exe`** (yes the space is required in this command and you may have your resource tools in a different location)*

It should tell you: [SC] CreateService SUCCESS

- *Run **`Regedit`**. Find: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\ClamD*
  - *Right click on ClamD and create a new Key*
  - *Name the key **`Parameters`***
  - *Click on the new Parameters Key, in the right pane create a new String Value*

- *Name the String Value* **`Application`**
- *Right click and Modify Application to:* **`C:\ClamAV\clamd.exe -c C:\ClamAV\clamd.conf`**
- *Close Regedit*

- *Run* **`services.msc`**
  - *Find ClamD in the list*
  - *Right click -> Properties*
  - *Change Startup type to: Automatic*
  - *Select the recovery tab and change all three failures to: Restart the Service*
  - *Select the General tab and Apply*

Freshclam Virus Signature Auto Update

Almost done!

- *Navigate to: Start->All Programs->Accessories->System Tools->Scheduled Tasks*
- *Add Scheduled Task*
- *Next->Browse to C:\ClamAV\freshclam.exe*
- *Select to Perform this task daily*
- *Next*
- *Select a good time to update*
- *Next*
- *Enter the user credentials to run as*
- *Next*
- *Select open Advance options*
- *Next*
- *Alter Run to be:* **`C\ClamAV\freshclam.exe -c C:\ClamAV\freshclam.conf`**
- *Select the Settings Tab*
- *Change the Stop Task to 30 minutes*
- *Apply*
- *Ok*

Right click and run the freshclam task to make sure you do not have any errors.

Go to ***Administration > Configuration parameters****:*

Field / Property	*Type*	*Description*
system.antivir	***String***	**`C:\ClamAV\clamscan.exe`**

background.antivirus.check	***Boolean***	`true`
background.antivirus.check.threads	***Integer***	`20`
background.antivirus.check.batch	***Integer***	`2`

More information about configuration parameters:

- *Antivirus configuration parameters*
- *Performance configuration parameters*

If everything was configured correctly, OpenKM will scan every uploaded document in order to check if it's safe.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Cleaning older activity log data

There's a table named OKM_ACTIVITY where openkm saves a log will the actions done by the users on all the nodes. In this table can be easily several milions of rows what can affect the perfomance of the database. Really you should be interested on most actual activity log data - for example last year - and keep the other data outside the database. For it is a good practice to perfom some cleaning process, exporting oldest activity log data and then deleting it from database.

Taking control of OKM_ACTIVITY log table size you can improve your database perfomance and reduce the total amount of your database backup size and time for doing it.

Doing backups of OKM_ACTIVITY by range and deleting from database you keep your activity log information safe without affecting database performance.

You always can restore the OKM_ACTIVITY logs in other database for doing forensic analytics.

MySQL

Linux

*Create a file named **clean_okm_activity.sh** at /root/ folder:*

```
#!/bin/sh
## BEGIN CONFIG ##
DATABASE_PASS="*secret*"
BACKUP_DIR="/mnt/backup/okm_activity"
## END CONFIG ##

mysqldump -h localhost -u root -p$DATABASE_PASS -w"ACT_DATE < (NOW() - INTERVAL 365 D
mysql -h localhost -u root -p$DATABASE_PASS okmdb < /root/delete.sql
```

*Make **clean_okm_activity.sh** as executable file:*

```
$chmod +x clean_okm_activity.sh
```

*Create a file named **delete.sql** at /root/ folder:*

```
DELETE FROM OKM_ACTIVITY WHERE ACT_DATE < (NOW() - INTERVAL 365 DAY);
ANALYZE TABLE OKM_ACTIVITY;
```

Configure crontab

To install the cron job, run:

```
$ sudo mkdir /root/logs

$ sudo crontab -e
```

And add these lines according to your personal configuration:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
MAILTO=nomail@openkm.com
@daily /root/clean_okm_activity.sh | tee /root/logs/okm_activity.$(date +\%Y.\%m.\%d_
```

More information at Crontab quick reference

*If you want to be notified by mail you should install "**postfix**" service in your server.*

Windows

*Create a file named **clean_okm_activity.bat** at c:\backup folder:*

```
set SQLFILE_DATE=%DATE:~6,4%.%DATE:~3,2%.%DATE:~0,2%
set SQLFILE_TIME=%TIME:~0,2%.%TIME:~3,2%
set SQLFILE=c:\backup\okm_activity-%SQLFILE_DATE%-%SQLFILE_TIME%.sql
set DATABASE_PASS=*secret*

mysqldump -h localhost -u root -p%DATABASE_PASS% -w"ACT_DATE < (NOW() - INTERVAL 365
mysql -h localhost -u root -p%DATABASE_PASS% okmdb < delete.sql
```

*Create a file named **delete.sql** at c:\backup folder:*

```
DELETE FROM OKM_ACTIVITY WHERE ACT_DATE < (NOW() - INTERVAL 365 DAY);
ANALYZE TABLE OKM_ACTIVITY;
```

You can use Windows Task Scheduler for configuring periodically your task.

Database replication with federated tables

The procedure for using FEDERATED tables is very simple. Normally, you have two servers running, either both on the same host or on different hosts. (It is possible for a FEDERATED table to use another table that is managed by the same server, although there is little point in doing so.)

You should have two database servers. Production MySQL server (A) and backup MySQL server (B).

First of all must enable federated tables on production server. Stop MySQL production service service and edit the MySQL configuration file (my.cnf or my.ini depending on your OS). Go to [mysqld] section and set below:

```
[mysqld]
# To enable federated feature
federated
```

Start your production MySQL service and execute from your preferred MySQL client tool the command:

```
show create table okm_activity:
```

You will get something like:

```
CREATE TABLE `okm_activity` (
  `ACT_ID` bigint(20) NOT NULL AUTO_INCREMENT,
  `ACT_ACTION` varchar(127) COLLATE utf8_bin DEFAULT NULL,
  `ACT_DATE` datetime DEFAULT NULL,
  `ACT_ITEM` varchar(64) COLLATE utf8_bin DEFAULT NULL,
  `ACT_PARAMS` varchar(4000) COLLATE utf8_bin DEFAULT NULL,
  `ACT_PATH` longtext COLLATE utf8_bin,
  `ACT_TENANT` bigint(20) DEFAULT NULL,
  `ACT_USER` varchar(64) COLLATE utf8_bin DEFAULT NULL,
  PRIMARY KEY (`ACT_ID`),
  KEY `IDX_ACTIVITY_TENANT` (`ACT_TENANT`),
  KEY `IDX_ACTIVITY_DATE` (`ACT_DATE`),
  KEY `IDX_ACTIVITY_TNTUSRACT` (`ACT_TENANT`,`ACT_USER`,`ACT_ACTION`),
  KEY `IDX_ACTIVITY_TNTDATACT` (`ACT_TENANT`,`ACT_DATE`,`ACT_ACTION`)
  ) ENGINE=innodb DEFAULT CHARSET=utf8 COLLATE=utf8_bin
```

*Create the table "**okm_activity**" in your backup MySQL server.*

*Create the federated table in your production MySQL server (take a look we have **changed the table name**, the **ENGINE=innodb for ENGINE=federated** and we have **added the connection link**):*

```
CREATE TABLE `okm_activity_remote` (
  `ACT_ID` bigint(20) NOT NULL AUTO_INCREMENT,
  `ACT_ACTION` varchar(127) COLLATE utf8_bin DEFAULT NULL,
  `ACT_DATE` datetime DEFAULT NULL,
  `ACT_ITEM` varchar(64) COLLATE utf8_bin DEFAULT NULL,
  `ACT_PARAMS` varchar(4000) COLLATE utf8_bin DEFAULT NULL,
  `ACT_PATH` longtext COLLATE utf8_bin,
  `ACT_TENANT` bigint(20) DEFAULT NULL,
  `ACT_USER` varchar(64) COLLATE utf8_bin DEFAULT NULL,
  PRIMARY KEY (`ACT_ID`),
  KEY `IDX_ACTIVITY_TENANT` (`ACT_TENANT`),
  KEY `IDX_ACTIVITY_DATE` (`ACT_DATE`),
  KEY `IDX_ACTIVITY_TNTUSRACT` (`ACT_TENANT`,`ACT_USER`,`ACT_ACTION`),
  KEY `IDX_ACTIVITY_TNTDATACT` (`ACT_TENANT`,`ACT_DATE`,`ACT_ACTION`)
  ) ENGINE=FEDERATED DEFAULT CHARSET=utf8 COLLATE=utf8_bin
  CONNECTION='mysql://user:poassword@server_b:3306/okmdb/okm_activity';
```

Finally you can periodically execute an script like this one from your MySQL production server:

```
select (NOW() - INTERVAL 1 DAY) into @dump_date;
insert into okm_activity_remote select * from okm_activity where act_date < @dump_dat
delete from okm_activity where act_date < @dump_date;
analyze table okm_activity;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Apache HTTP Reverse-Proxy

Exposing OpenKM directly from Tomcat can be risky, if you need the application to be accessed from the Internet (for example https://issues.jboss.org/browse/JBAS-3861). It could be infected by PerlBot. 8080 may be blocked by a firewall, for this reason is a recommendable to expose your OpenKM installation through the standard web port 80. In the following steps we explain how to configure Apache to handle these requests and forward to Tomcat application server using the AJP13 protocol.

From the Apache documentation: The AJP13 protocol is packet-oriented. A binary format was presumably chosen over the more readable plain text for performance reasons. The web server communicates with the servlet container over TCP connections. To cut down on the expensive process of socket creation, the web server will attempt to maintain persistent TCP connections to the servlet container, and to reuse a connection for multiple request/response cycles.

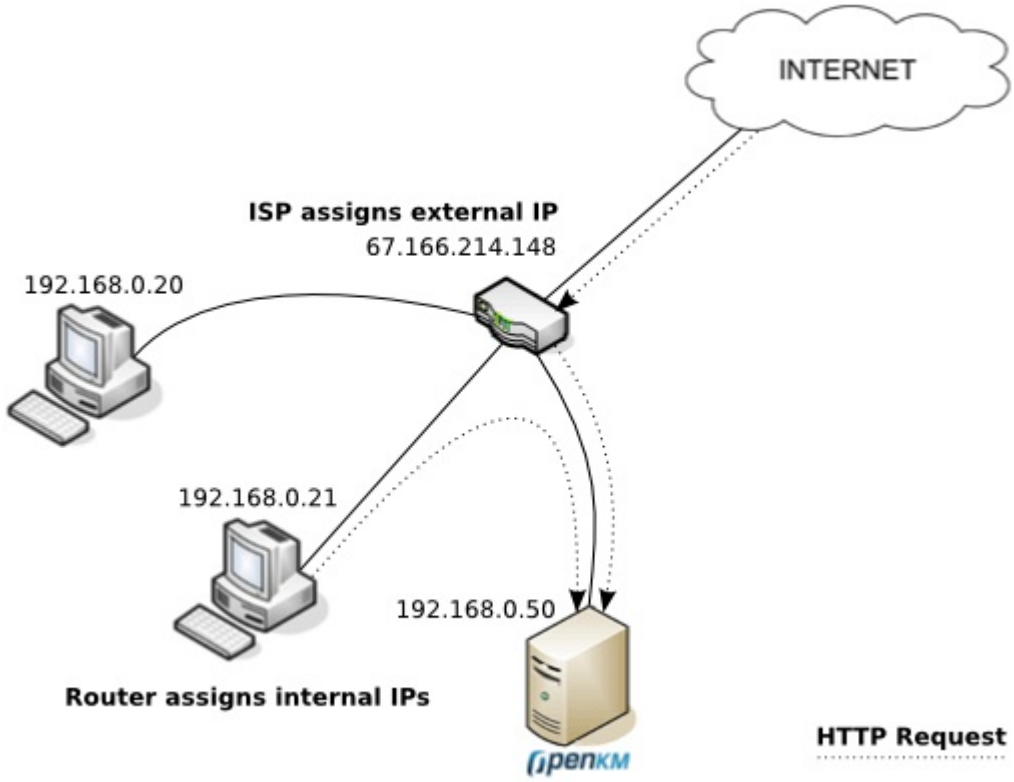
Internal IP vs external IP

Your OpenKM can be accessed from two different zones: Internet and LAN. This means that to access this server you need to use two IPs: external IP (Internet) and internal IP (LAN).

The internal IP address (also known as "local IP address") is the address that is assigned by your local network router that often begins with 192.168.x.x. These IP addresses can only be seen by other computers in your local network (LAN) and not by any computers connected in an external network such the Internet.

To reach the Internet or a computer in another network your computer is often assigned an external IP address, which can then be used to refer to the computer in your local network.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



In the above picture, there are three computers in the local network that have each been assigned their own internal IP address by the router. The ISP is connected to the router and gives the router an external IP address that allows it to communicate with the Internet. On the Internet everyone sees your external IP address, but any information coming from the router is "converted" from the external IP address to the internal IP address.

So if you want your OpenKM installation accessible from both LAN and Internet, the trick here is to configure the client computers to resolve your ***internal IP*** *(192.168.0.50) if they are inside the LAN or the* ***external IP*** *if they are on Internet. To resolve the Internet IP (67.166.214.148) your computer uses the public DNS. So, you need to configure a sort of DNS server inside the LAN or modify every client host file to resolve to the internal IP.*

Debian and Ubuntu

The first thing is to install the required Apache software. From Debian / Ubuntu you can install Apache with a single command:

```
$ sudo apt-get install apache2
```

Edit the file named ***/etc/apache2/apache2.conf*** *and configure a ServerName to prevent warnings in the Apache startup process:*

```
$ vim /etc/apache2/apache2.conf
```

```
ServerRoot "/etc/apache2"
ServerName "your-domain.com"
```

Edit apache2.conf is optional.

Enable the proxy module:

```
$ sudo a2enmod proxy_ajp
$ sudo a2enmod proxy_wstunnel
```

To get websocket working you also need Tomcat 7.0.61 or later. Note that ***proxy_wstunnel*** *module requires Apache 2.4.5 or later. Please refer to Apache module documentation.*

Create the configuration file ***/etc/apache2/sites-available/openkm.conf*** *with this content:*

```
$ vim /etc/apache2/sites-available/openkm.conf
```

```
<VirtualHost *:80>
  ServerName openkm.your-domain.com
  RedirectMatch ^/$ /openkm

  # WebSocket support - needs proxy-wstunnel
  ProxyPass /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/webSocket
  ProxyPassReverse /OpenKM/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/web

  ProxyPass /openkm ajp://127.0.0.1:8009/openkm keepalive=On
  ProxyPassReverse /openkm http://openkm.your-domain.com/openkm

  ErrorLog /var/log/apache2/openkm-error.log
  CustomLog /var/log/apache2/openkm-access.log combined
</VirtualHost>
```

Change openkm.your-domain.com with your server IP or your domain value.

The VirtualHost ServerName must be other than ServerName in the main Apache configuration. Enable this site configuration:

```
$ cd /etc/apache2/sites-available/
$ sudo a2ensite openkm.conf
```

You have to enable explicitly the proxy access editing the Apache configuration file ***/etc/apache2/mods-available/proxy.conf****:*

```
$ vim /etc/apache2/mods-available/proxy.conf
```

```
<IfModule mod_proxy.c>
  #turning ProxyRequests on and allowing proxying from all may allow
  #spammers to use your proxy to send email.

  ProxyRequests Off

  <Proxy *>
    AddDefaultCharset off
    Order deny,allow
    Allow from all
```

```
    Deny from all
    #Allow from .example.com
  </Proxy>

  # Enable/disable the handling of HTTP/1.1 "Via:" headers.
  # ("Full" adds the server version; "Block" removes all outgoing Via: headers)
  # Set to one of: Off | On | Full | Block

  ProxyVia On
</IfModule>
```

For security reasons, might you be interested in removing default apache configuration:

```
$ a2dissite 000-default.conf
```

Check the configuration

Restart Apache:

```
$ sudo /etc/init.d/apache2 restart
```

Check the access your OpenKM installation from http://openkm.your-domain.com/.

Another advantage of using Apache is that you can log OpenKM access and generate web statistics.

To check the configuration files can execute the command:

```
/usr/sbin/apache2 -t
```

Red Hat and CentOS

Use the yum application manager to install Apache:

```
$ sudo yum install httpd
```

Enable it at boot:

```
$ sudo chkconfig httpd --level 2345 on
```

*Now create the file **/etc/httpd/conf.d/openkm.conf** with this content:*

```
$ vim /etc/httpd/conf.d/openkm.conf
```

```
<VirtualHost *:80>
  ServerName openkm.your-domain.com
  RedirectMatch ^/$ /openkm

  # WebSocket support
  ProxyPass /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/webSocket
  ProxyPassReverse /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/web
```

```
  ProxyPass /openkm ajp://127.0.0.1:8009/openkm keepalive=On
  ProxyPassReverse /openkm http://openkm.your-domain.com/openkm

  ErrorLog /var/log/httpd/openkm-error.log
  CustomLog /var/log/httpd/openkm-access.log combined
</VirtualHost>
```

Change openkm.your-domain.com with your server IP or your domain value.

Check the configuration

Restart Apache:

```
$ sudo /etc/init.d/httpd restart
```

Check the access your OpenKM installation from http://openkm.your-domain.com/.

Another advantage of using Apache is that you can log OpenKM access and generate web statistics.

Windows

Install de apache webserver from http://httpd.apache.org/download.cgi (windows binary no mod_ssl)

*Edit the **C:Program files\Apache Software Foundation\Apache2.2\conf\httpd.conf***

```
# Enable proxy modules
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so

#Add in bottom
NameVirtualHost *:80
<VirtualHost *:80>
  ServerName openkm.your-domain.com
  RedirectMatch ^/$ /openkm

  ProxyRequests Off
  ProxyVia On

  <Proxy *>
    AddDefaultCharset off
    Order deny,allow
    Allow from all
    Deny from All
  </Proxy>

  # WebSocket support
  ProxyPass /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/webSocket
  ProxyPassReverse /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/web

  ProxyPass /openkm ajp://127.0.0.1:8009/openkm keepalive=On
  ProxyPassReverse /openkm http://openkm.your-domain.com/openkm

  ErrorLog logs/openkm-error_log
  CustomLog logs/openkm-access_log common
</VirtualHost>
```

Change openkm.your-domain.com with your server ip or your domain value.

Max OS X

Edit the file called ***/etc/apache2/apache2.conf*** *and configure a* ***ServerName***, ***enable proxy modules*** *and* ***mod_proxy****:*

```
ServerRoot "/usr"
ServerName "openkm.your-domain.com"

LoadModule proxy_module libexec/apache2/mod_proxy.so
LoadModule proxy_ajp_module libexec/apache2/mod_proxy_ajp.so

<IfModule mod_proxy.c>
  ProxyRequests Off
  <Proxy "*">
    AddDefaultCharset off
    Allow from all
    Deny from all
    Order Deny,Allow
  </Proxy>
  ProxyVia On
</IfModule>
```

Change openkm.your-domain.com with your server IP or your domain value.

Now create the configuration file ***/etc/apache2/sites/openkm.conf*** *with this content:*

```
<VirtualHost *:80>
  ServerName openkm.your-domain.com
  RedirectMatch ^/$ /openkm

  # WebSocket support
  ProxyPass /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/webSocket
  ProxyPassReverse /openkm/frontend/webSocket ws://127.0.0.1:8080/openkm/frontend/web

  ProxyPass /openkm ajp://127.0.0.1:8009/openkm keepalive=On
  ProxyPassReverse /openkm http://openkm.your-domain.com/openkm

  ErrorLog /var/log/apache2/openkm-error.log
  CustomLog /var/log/apache2/openkm-access.log combined
</VirtualHost>
```

Check the configuration

Restart Apache:

Check the access to your OpenKM installation from [*http://openkm.your-domain.com/*](http://openkm.your-domain.com/)*.*

Another advantage of using Apache is that you can log OpenKM access and generate web statistics.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Troubleshooting

If you see an error message like:

Invalid command 'RewriteEngine', perhaps misspelled or defined by a module not included in the server configuration

You need to enable this Apache module:

```
$ sudo a2enmod rewrite
$ sudo a2enmod proxy_http
$ sudo a2enmod headers
```

Additional information

- *Apache Module mod_proxy*
- *Apache Module mod_proxy_ajp*
- *Apache Module mod_rewrite*
- *mod_rewrite, a beginner's guide*
- *Using Apache Virtual Hosts and ProxyPass Together*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Apache HTTP for changing context URL

First of all you must have one of these configurations running on your server:

- *Configuring Apache HTTP Reverse-Proxy*
- *Configuring Apache HTTPS Reverse-Proxy*

On the sample below the http://openkm.your-domain.com/openkm/frontend has been rewrited to http://openkm.yourdomain.com/frontend

Enable the headers module:

```
$ sudo a2enmod headers
```

To change /OpenKM application context try this as your VirtualHost configuration:

```
<VirtualHost *:80>
  ServerName openkm.your-domain.com

  ProxyPass /frontend/webSocket ws://localhost:8080/OpenKM/frontend/webSocket
  ProxyPassReverse /frontend/webSocket ws://localhost:8080/OpenKM/frontend/webSocket

  ProxyPass / ajp://localhost:8009/openkm/
  ProxyPassReverse / http://openkm.your-domain.com/openkm/

  RequestHeader edit X-GWT-Module-Base ^(http://openkm.your-domain.com)/frontend/(.*)
  ProxyPassReverseCookiePath /openkm /

  ErrorLog /var/log/apache2/openkm-error.log
  CustomLog /var/log/apache2/openkm-access.log combined
</VirtualHost>
```

*You need also set the configuration parameter **system.apache.request.header.fix** to on (true).*

Additional information

- *Apache configuration for GWT applications*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Apache HTTPS Reverse-Proxy

Exposing OpenKM directly from Tomcat can be dangerous if you need the application to be accessed from Internet (for example https://issues.jboss.org/browse/JBAS-3861). As result you can be infected by PerlBot. Also this 8080 may be blocked by a firewall. For these reasons, it is a good practice to expose your OpenKM installation through the standard web port 80. In the following steps we explain how to configure Apache to handle these request and forward to Tomcat application server using the AJP13 protocol.

From the Apache documentation: The AJP13 protocol is packet-oriented. A binary format was presumably chosen over the more readable plain text for performance reasons. The web server communicates with the servlet container over TCP connections. To cut down on the expensive process of socket creation, the web server will attempt to maintain persistent TCP connections to the servlet container, and to reuse a connection for multiple request/response cycles.

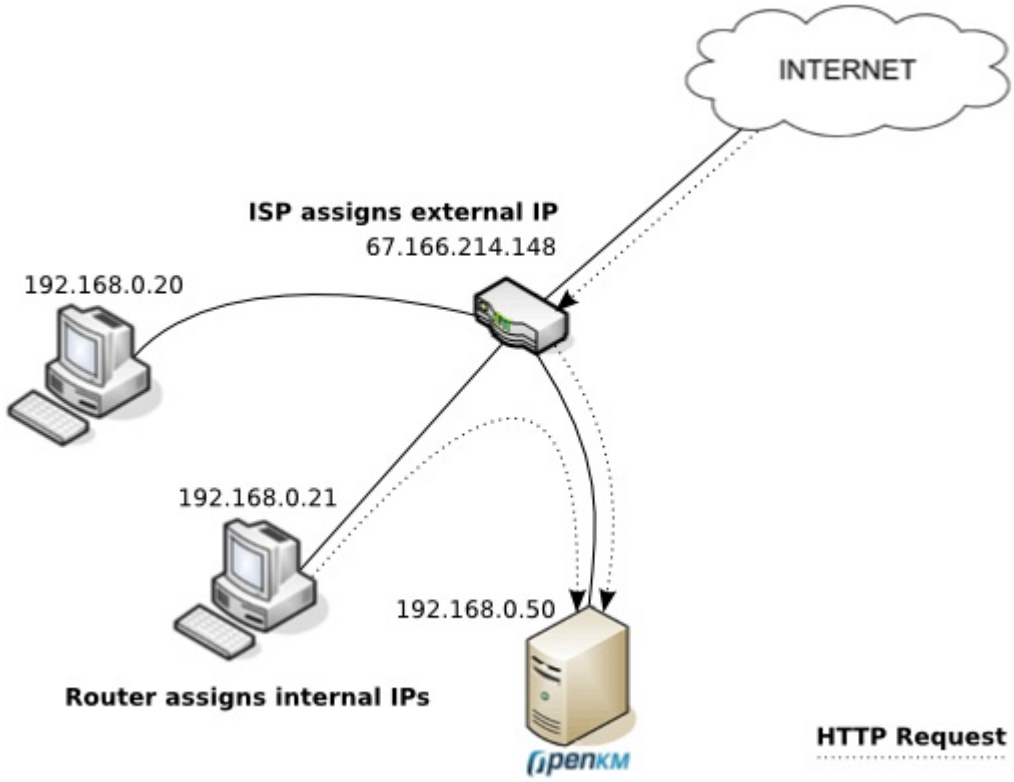
Internal IP vs external IP

Your OpenKM can be accessed from two different zones: Internet and LAN. This means that to access this server you need to use two IPs: external IP (Internet) and internal IP (LAN).

The internal IP address (also known as "local IP address") is the address that is assigned by your local network router that often begins with 192.168.x.x. These IP addresses can only be seen by other computers in your local network (LAN) and not by any computer connected in an external network such the Internet.

To reach the Internet or a computer in another network your computer is often assigned an external IP address, which can then be used to refer to the computer in your local network.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



In the above picture, there are three computers in the local network that have each been assigned their own internal IP address by the router. The ISP is connected to the router and gives the router an external IP address that allows it to communicate with the Internet. On the Internet everyone sees your external IP address, but any information coming from the router is "converted" from the external IP address to the internal IP address.

So if you want your OpenKM installation to be accessible from both LAN and Internet, the trick here is to configure the client computers to resolve your ***internal IP*** *(192.168.0.50) if they are inside the LAN or the* ***external IP*** *if they are on Internet. To resolve the Internet IP (67.166.214.148) your computer uses the public DNS. So, you need to configure a sort of DNS server inside the LAN or modify every client host file to resolve to the internal IP.*

Debian and Ubuntu

The first thing in to install the required Apache software. From Debian / Ubuntu you can install Apache with a single command:

```
$ sudo apt-get install apache2
```

Edit the file named ***/etc/apache2/apache2.conf*** *and configure a ServerName to prevent warnings in the Apache startup process:*

```
$ vim /etc/apache2/apache2.conf
```

```
ServerRoot "/etc/apache2"
ServerName "your-domain.com"
```

Edit apache2.conf is optional.

Enable the proxy module:

```
$ sudo a2enmod proxy_ajp
$ sudo a2enmod proxy_wstunnel
```

*To get websocket working you also need Tomcat 7.0.61 or later. Note that **proxy_wstunnel** module requires Apache 2.4.5 or later. Please refer to Apache module documentation.*

Create the Certificates

```
$ sudo mkdir /etc/apache2/ssl
$ sudo /usr/sbin/make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/apache2/ssl/apache.pem
$ sudo a2enmod ssl
```

If you need to convert .pem file to crt use the command:

```
$ openssl x509 -in cacert.pem -out cacert.crt
```

Also you could be interested on Configuring OpenSSL.

*Ensure ports **443** is listen in **/etc/apache2/ports.conf**:*

```
$ cat /etc/apache2/ports.conf | grep Listen
```

*Create the configuration file **/etc/apache2/sites-available/openkm.conf** with this content:*

```
$ vim /etc/apache2/sites-available/openkm.conf
```

```
<VirtualHost *:443>
  ServerName openkm.your-domain.com
  RedirectMatch ^/$ /openkm

  # WebSocket support - needs proxy-wstunnel
  ProxyPass /openkm/frontend/webSocket wss://127.0.0.1:8080/openkm/frontend/webSocket
  ProxyPassReverse /openkm/frontend/webSocket wss://127.0.0.1:8080/openkm/frontend/we

  ProxyPass /openkm ajp://127.0.0.1:8009/openkm
  ProxyPassReverse /openkm https://openkm.your-domain.com/openkm

  ErrorLog /var/log/apache2/openkm.your-domain.com-error.log
  CustomLog /var/log/apache2/openkm.your-domain.com-access.log combined

  SSLEngine on
  SSLCertificateFile /etc/apache2/ssl/apache.pem
</VirtualHost>
```

Change openkm.your-domain.com with your server IP or your domain value.

The VirtualHost ServerName must be other than ServerName in the main Apache configuration. Enable this site configuration:

```
$ cd /etc/apache2/sites-available/
$ sudo a2ensite openkm.conf
```

Redirect all HTTP connections to HTTPS

This step is optional.

Edit the configuration file ***/etc/apache2/sites-available/openkm.conf*** *and add these content*

```
$ vim /etc/apache2/sites-available/openkm.conf
```

```
<VirtualHost *:80>
  ServerName openkm.your-domain.com
  Redirect permanent / https://openkm.your-domain.com/
</VirtualHost>
```

Check the configuration

Restart Apache:

```
$ sudo /etc/init.d/apache2 restart
```

Check the access your OpenKM installation from https://openkm.your-domain.com/.

Another advantage of using Apache is that you can log OpenKM access and generate web statistics.

To check the configuration files can execute the command:

```
$ /usr/sbin/apache2 -t
```

Red Hat and CentOS

Use the yum application manager to install Apache:

```
$ sudo yum install httpd
```

Enable it at boot:

```
$ sudo chkconfig httpd --level 2345 on
```

Create the Certificate

```
$ sudo yum install mod_ssl openssl crypto-utils
```

Generate private keys (for more information visit Apache HTTP Secure Server Configuration)

```
$ genkey your-domain.com
```

Usually you want to generate self-certificate and not sending it to Certify Authority. Is good practice to set a password on private key, but in this case each time you restart Apache service it'll be demanded.

Also you could be interested on Configuring OpenSSL.

During the process will be generated two files at:

```
ReplaceSSLCertificateFile /etc/pki/tls/certs/your-domain.com.cert

SSLCertificateKeyFile /etc/pki/tls/private/your-domain.com.key
```

*Now create the file **/etc/httpd/conf.d/openkm.conf** with this content:*

```
$ vim /etc/httpd/conf.d/openkm.conf
```

```
<VirtualHost *:443>
  ServerName openkm.your-domain.com
  ProxyPass / ajp://localhost:8009/openkm/
  ProxyPassReverse / http://your-domain.com/openkm/
  ErrorLog /var/log/your-domain.com-error.log
  CustomLog /var/log/your-domain.com-access.log combined

  SSLEngine on
  SSLCertificateFile /etc/pki/tls/certs/your-domain.com.cert
  SSLCertificateKeyFile /etc/pki/tls/private/your-domain.com.key
</VirtualHost>
```

Change openkm.your-domain.com with your server ip or your domain value.

*Finally must modify **SSLCertificateFile** and **SSLCertificateKeyFile** values in file **/etc/httpd/conf.d/ssl.conf**:*

```
$ vim /etc/httpd/conf.d/ssl.conf
```

```
SSLCertificateFile /etc/pki/tls/certs/your-domain.com.cert
SSLCertificateKeyFile /etc/pki/tls/private/your-domain.com.key
```

Check the configuration

Restart Apache:

```
$ sudo /etc/init.d/httpd restart
```

Check the access your OpenKM installation from https://openkm.your-domain.com/.

Another advantage of using Apache is that you can log OpenKM access and generate web statistics.

Troubleshooting

If you see an error like:

Invalid command 'RewriteEngine', perhaps misspelled or defined by a module not included in the server configuration

You need to enable this Apache module:

```
$ sudo a2enmod rewrite
$ sudo a2enmod proxy_http
$ sudo a2enmod headers
```

Additional information

- *Apache Module mod_proxy*
- *Apache Module mod_proxy_ajp*
- *Apache Module mod_rewrite*
- *mod_rewrite, a beginner's guide*
- *Using Apache Virtual Hosts and ProxyPass Together*
- *Rewrite HTTP To HTTPS*
- *Redirect to SSL*
- *Configuración SSL de Tomcat*
- *How To Create a SSL Certificate on Apache for CentOS 6*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Nginx HTTP Reverse-Proxy

Exposing OpenKM directly from Tomcat can be dangerous if you need the application to be accessed from the Internet (for example https://issues.jboss.org/browse/JBAS-3861). As result it can be infected by PerlBot. Also this 8080 may be blocked by a firewall. For these reasons, is a good idea to expose your OpenKM installation through the standard web port 80. In the following steps we explain how to configure Apache to handle these requests and forward them to Tomcat application server using the AJP13 protocol.

From the Apache documentation: The AJP13 protocol is packet-oriented. A binary format was presumably chosen over the more readable plain text for reasons of performance. The web server communicates with the servlet container over TCP connections. To cut down on the expensive process of socket creation, the web server will attempt to maintain persistent TCP connections to the servlet container, and to reuse a connection for multiple request/response cycles.

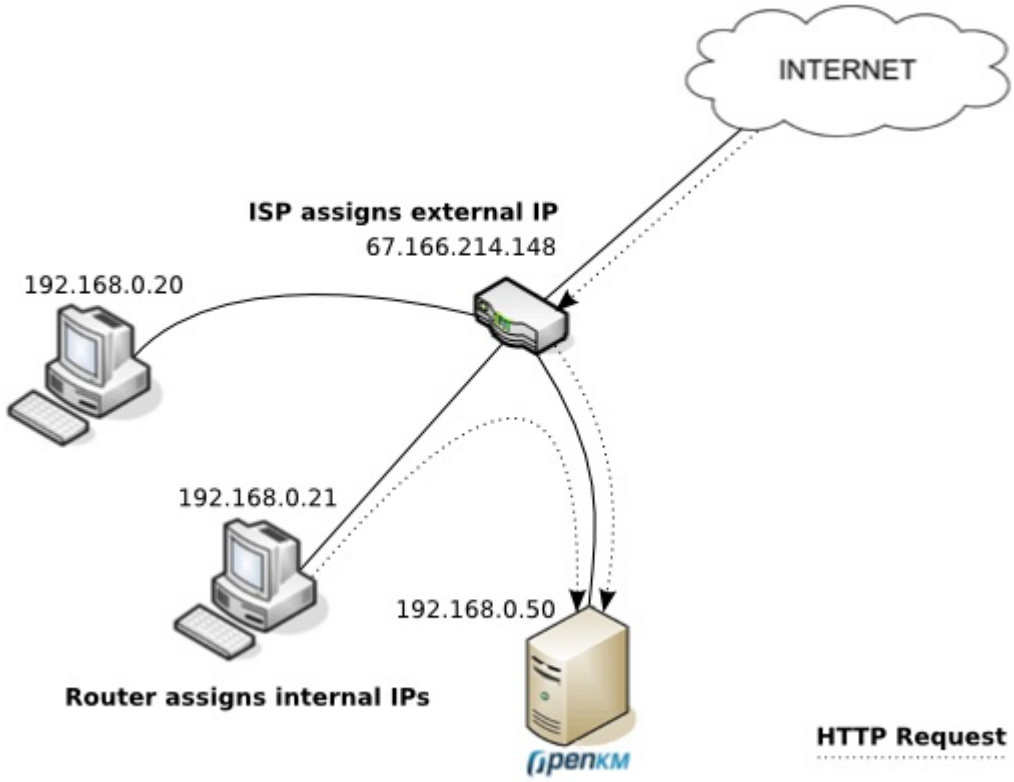
Internal IP vs external IP

Your OpenKM can be accessed from two different zones: Internet and LAN. This means that to access this server you need to use two IPs: external IP (Internet) and internal IP (LAN).

The internal IP address (also known as "local IP address") is the address that is assigned by your local network router that often begins with 192.168.x.x. These IP addresses can only be seen by other computers in your local network (LAN) and not by any computers connected in an external network such the Internet.

To reach the Internet or a computer in another network your computer is often assigned an external IP address, which can then be used to refer to the computer in your local network.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



In the above picture, there are three computers in the local network that have each been assigned their own internal IP address by the router. The ISP is connected to the router and gives the router an external IP address that allows it to communicate with the Internet. On the Internet everyone sees your external IP address, but any information coming from the router is "converted" from the external IP address to the internal IP address.

So if you want your OpenKM installation accessible from both LAN and Internet, the trick here is configure the client computers to resolve your ***internal IP*** *(192.168.0.50) if they are inside the LAN or the* ***external IP*** *if they are on the Internet. To resolve the Internet IP (67.166.214.148) your computer uses the public DNS. So, you need to configure a sort of DNS server inside the LAN or modify every client host file to resolve to the internal IP.*

Debian and Ubuntu

The first thing in to install the required Apache software. From Debian / Ubuntu you can install Apache with a single command:

```
$ sudo apt-get install nginx
```

Now create the configuration file ***/etc/nginx/sites-available/openkm*** *with this content:*

```
$ vim /etc/nginx/sites-available/openkm
```

```
server {
    listen 80;
    server_name openkm.your-domain.com;
    # Avoid checking files size #
```

```
    client_max_body_size 0;

    rewrite ^/$ /openkm permanent;

    location /openkm/frontend/webSocket {
        proxy_pass http://localhost:8080/openkm/frontend/webSocket;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }

    location /openkm {
         proxy_set_header X-Forwarded-Host $host;
         proxy_set_header X-Forwarded-Server $host;
         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
         proxy_pass http://localhost:8080/openkm;
    }
}
```

Change openkm.your-domain.com with your server ip or your domain value.

Enable this site configuration:

```
$ ln -s /etc/nginx/sites-available/openkm /etc/nginx/sites-enabled/
```

Check the configuration

Restart Nginx:

```
$ sudo /etc/init.d/nginx restart
```

Check the access your OpenKM installation from http://openkm.your-domain.com/.

Another advantage of using Nginx is that you can log OpenKM access and generate web statistics.

Red Hat and CentOS

Use the yum application manager to install Nginx:

```
$ sudo yum install nginx
```

Enable it at boot:

```
$ sudo chkconfig nginx --level 2345 on
```

*Depending on your CentOS version this Nginx version may be too old so it's better to install it from their repositories. To do this, create a file called **/etc/yum.repos.d/nginx.repo** and include one on these configurations:*

CentOS

```
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/centos/$releasever/$basearch/
gpgcheck=0
enabled=1
```

RedHat

```
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/rhel/$releasever/$basearch/
gpgcheck=0
enabled=1
```

Now create the configuration file /etc/nginx/conf.d/openkm.conf with this content:

```
$ vim /etc/nginx/conf.d/openkm.conf
```

```
server {
    listen 80;
    server_name openkm.your-domain.com;
    # Avoid checking files size #
    client_max_body_size 0;

    rewrite ^/$ /openkm/ permanent;

    location /openkm/frontend/webSocket {
        proxy_pass http://localhost:8080/openkm/frontend/webSocket;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
    }

    location /openkm {
         proxy_set_header X-Forwarded-Host $host;
         proxy_set_header X-Forwarded-Server $host;
         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
         proxy_pass http://localhost:8080/openkm;
    }
}
```

Change openkm.your-domain.com with your server IP or your domain value.

Check the configuration

Restart Nginx:

```
$ sudo /etc/init.d/nginx restart
```

Check the access your OpenKM installation from http://openkm.your-domain.com/.

If you can't access server port from other machines, please take a look at your firewall configuration. For more info read [How

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

To Set Up a Basic Iptables Firewall on Centos 6.

Another advantage of using Nginx is that you can log OpenKM access and generate web statistics.

In case you see this kind of errors in the Nginx log:

```
connect() to 127.0.0.1:8080 failed (13: Permission denied)
```

Execute this command and try again:

```
$ setsebool -P httpd_can_network_connect on
```

These are several useful command related to SELinux:

Get all related HTTP configuration

```
$ getsebool -a | grep httpd
```

More information at SELinux Booleans.

Configuring Lucene Analyzer

Depending on the language used in the documents and properties, you have obtain better search results configuring a proper Lucene Analyzer.

By default, OpenKM use the ***org.apache.lucene.analysis.standard.StandardAnalyzer*** *which works fine with English and most languages, but you can get better search results configuring more specific analyzer for you language.*

Some analyzers:

- *org.apache.lucene.analysis.en.EnglishAnalyzer*
- *org.apache.lucene.analysis.es.SpanishAnalyzer*
- *org.apache.lucene.analysis.fr.FrenchAnalyzer*
- *org.apache.lucene.analysis.it.ItalianAnalyzer*
- *org.apache.lucene.analysis.de.GermanAnalyzer*
- *org.apache.lucene.analysis.el.GreekAnalyzer*
- *org.apache.lucene.analysis.hi.HindiAnalyzer*

Special analyzers:

- *com.openkm.analysis.AccentInsensitiveAnalyzer*

More information at:

- *Analyzer*
- *ReusableAnalyzerBase*
- *StopwordAnalyzerBase*

If you want non accent sensitive analyzer can try with AccentInsensitiveAnalyzer.

If you are working with oriental languages like Chinese or Japanese you have several analyzers to use. Read Lucene documentation. You can also try ik-analyzer.

If you want only a white space tokenized analyzer can try with this one WhitespaceAnalyzer.

If you have not configured the search analyzer before the first time you start OpenKM, then Lucene indexed will be created using this default analyzer.

If you want to change this configuration property after the OpenKM repository has been created you need to Rebuild Lucene Indexes.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Once the operation has been completed, the Lucene indexes will be using the new analyzer.

For more information take a look at Rebuild indexes.

Configure an Analyzer

Edit the $TOMCAT_HOME/openkm.properties file and add the line:

```
hibernate.search.analyser=org.apache.lucene.analysis.es.SpanishAnalyzer
```

The changes will take effect after restarting the application.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring OCR engine

OpenKM can work with several OCR engines, for example Tesseract 2.x, Tesseract 3.x, Cuneiform or Abby among others.

OpenKM can be integrated with any OCR engine that can be executed from command line.

Tesseract

Tesseract is an Open Source OCR engine adopted by Goggle. It works really well. The OCR natively can read TIFF documents and has hight ratio of recognition with images 300 dpi of resolution and converted to lineart (1 bit color).

The supported image formats are:

- *TIFF*
- *PNG*
- *JPG*
- *GIF*

Installation

OS	Description
Ubuntu and Debian	*$ apt-get install tesseract-ocr tesseract-ocr-eng* *Depending on your locale you would like to install other language file.* *More information at:* • *https://github.com/tesseract-ocr* • *https://code.google.com/p/tesseract-ocr/*
Red Hat and CentOS	*More information at:* • *https://code.google.com/p/tesseract-ocr/wiki/Compiling* • *https://code.google.com/p/python-tesseract/wiki/HowToCompileForCentos* • *http://www.vicchiam.com/blog/?p=168*
Windows	*Download from https://code.google.com/p/tesseract-ocr/ and follow the installation wizard.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Check installation

From command line execute (where image.jpg should be an existing image file):

```
$ tesseract image.jpg text
```

*The file **image.jpg** is an image file.*

*If all goes right a file named **text.txt** what contains the extracted text will be created.*

Configuration

*Go to **Administration** > Configuration parameters and edit the parameter named **system.ocr**:*

OS	

Linux	`/usr/bin/tesseract ${fileIn} ${fileOut}`
	or
	`/usr/bin/tesseract ${fileIn} ${fileOut} -l esp`
	*The parameter **-l esp** indicate must be used the Spanish language support while executing OCR.*
	To see all tesseract options execute:
	`$ tesseract`

Windows	`c:\Tesseract-ocr-3.0.2\tesseract.exe ${fileIn} ${fileOut}`
	or
	`c:\Tesseract-ocr-3.0.2\tesseract.exe ${fileIn} ${fileOut} -l spa`
	*The parameter **-l esp** indicate must be used the Spanish language support while executing OCR.*
	To see all tesseract options execute:
	`c:\Tesseract-ocr-3.0.2> tesseract.exe`

*Parameter **${fileIn}** will be replaced internally by image document to be processed.*

*Parameter **${fileOut}** will be replaced internally by temporal text file.*

*Can set other additional parameters like **-l spa** parameter in example.*

Each parameter is separated by a single "white space".

Additional information

- *http://code.google.com/p/tesseract-ocr/*
- *Tesseract - Summary & first experiences*.
- *Tesseract OCR Google Groups*
- *First Interactions with Tesseract OCR on Ubuntu Linux*
- *http://code.google.com/p/tesseract-ocr/wiki/ReadMe*
- *http://code.google.com/p/tesseract-ocr/wiki/FAQ*

Cuneiform

We recommend the use of Tesseract rather than cuneiform.

The supported image formats are:

- *TIFF*
- *PNG*
- *JPG*
- *GIF*

Installation

OS	*Description*
Ubuntu and Debian	*$ apt-get install cuneiform* *More information at:* • *http://cognitiveforms.com/products_and_services/cuneiform*
Red Hat and Centos	*More information at:* • *http://cognitiveforms.com/products_and_services/cuneiform*
Windows	*Not available to be executed from command line.* *More information at:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- http://cognitiveforms.com/products_and_services/cuneiform

Check installation

From command line execute (where image.jpg should be an existing image file):

```
$ cuneiform image.jpg -o text.txt
```

*The file **image.jpg** is an image file.*

*If all goes right a file named **text.txt** what contains the extracted text will be created.*

Configuration

*Go to **Administration** > Configuration parameters and edit the parameter named **system.ocr**:*

OS	

Linux	`/usr/bin/cuneiform ${fileIn} -o ${fileOut}`
	or
	`/usr/bin/cuneiform ${fileIn} -o ${fileOut} -l esp`
	*The parameter **-l esp** indicate must be used the Spanish language support while executing OCR.*
	To see all cuneiform options execute:
	`$ cuneiform`

Windows	*Not available.*

*Parameter **${fileIn}** will be replaced internally by image document to be processed.*

*Parameter **${fileOut}** will be replaced internally by temporal text file.*

*Can set other additional parameters like **-l spa** parameter in example.*

*Before the **${fileOut}** must be set **-o** parameter.*

Each parameter is separated by a single "white space".

Abby OCR for linux

OCR4 Linux is a commercial OCR engine. It is the slowest of all tested tools, but keep in mind that it also reads nearly any image format, while you probably need to convert your images for the other tools first.

Installation

OS	Description
Ubuntu, Debian, Red Hat and Centos	*More information at:* • *http://www.ocr4linux.com/*
Windows	*Not available.*

Check installation

From command line execute (where image.jpg should be an existing image file):

```
$ /usr/local/bin/abbyyocr9 abbyyocr11 -ii -fm -if image.jpg -tet UTF8 -of text.txt
```

*The file **image.jpg** is an image file.*

*If all goes right will be created a file named **text.txt** what contains the extracted text.*

Configuration

*Go to **Administration** > Configuration parameters and edit the parameter named **system.ocr**:*

OS	

Linux	`/usr/local/bin/abbyyocr11 -ii -fm -if ${fileIn} -tet UTF8 -of ${fileOut}`
	There're a lot of parameters, to see all ocr4linux options execute:
	`$ abbyyocr11 -?`

Windows	*Not available.*

*Parameter **${fileIn}** will be replaced internally by image document to be processed.*

*Parameter **${fileOut}** will be replaced internally by temporal text file.*

*Can set other additional parameters like **-l spa** parameter in example.*

Each parameter is separated by a single "white space".

Dictionary

You can also use an OpenOffice.org dictionary to enhance the OCR process. Can find these language specific dictionaries at OpenOffice.org Dictionary Repository.

Configuration

*Go to **Administration** > Configuration parameters and edit the parameter named **system.openoffice.dictionary**:*

OS	
Linux	`/home/openkm/dictionary/en-GB.zip`
Windows	`c:\tomcat-7.0.61\dictionary\en-GB.zip`

Dictionaries comes in .zip or oxt format. Can use both.

OCR rotate configuration

There's a configuration parameter that force document ocr on rotated document.

The feature is useful for example in upside down scanned pages, where standard OCR direction will fails.

When the feature is enabled the operation will be done on all OCR extractions, that can take undesired effects on OCR templates feature.

Configuration

*Go to **Administration** > Configuration parameters and edit the parameter named **system.ocr.rotate**:*

Property	Description
system.ocr.rotate	*The parameter is a collection of degrees separated by character ";"* `90;180;270;`

Additional information:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Linux OCR Software Comparison*
- *Tesseract OCR*
- *Cuneiform*
- *Ocropy*
- *Gocr*
- *Ocrad*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring OpenSSL

Create OpenSSL configuration file

This sample has been done with Ubuntu and Debian server, for Red Hat and CentOS the path of Apache files is not the same.

*On Ubuntu and Debian the Apache path **/etc/apache2** is **/etc/httpd** on Red Hat and CentOS.*

*Edit file **/etc/ssl/openssl.cnf***

```
$ vim /etc/ssl/openssl.cnf
```

```
RANDFILE          = $ENV::SSLDIR/.rnd

[ ca ]
default_ca        = CA_default

[ CA_default ]
dir               = /opt/ca
certs             = $dir/certs
new_certs_dir     = $dir/newcerts
crl_dir           = $dir/crl
database          = $dir/index.txt
private_key       = $dir/private/ca.key
certificate       = $dir/ca.crt
serial            = $dir/serial
crl               = $dir/crl.pem
RANDFILE          = $dir/private/.rand
default_days      = 365
default_crl_days  = 30
default_md        = sha1
preserve          = no
policy            = policy_anything
name_opt          = ca_default
cert_opt          = ca_default

[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ req ]
default_bits            = 2048
default_md              = sha1
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
x509_extensions         = v3_ca
string_mask             = nombstr

[ req_distinguished_name ]
countryName             = Country Name (2 letter code)
countryName_min         = 2
countryName_max         = 2
stateOrProvinceName     = State or Province Name (full name)
```

```
localityName              = Locality Name (eg, city)
0.organizationName        = Organization Name (eg, company)
organizationalUnitName    = Organizational Unit Name (eg, section)
commonName                = Common Name (eg, YOUR name)
commonName_max            = 64
emailAddress              = Email Address
emailAddress_max          = 64

[ usr_cert ]
basicConstraints          = CA:FALSE
# nsCaRevocationUrl         = https://url-to-exposed-clr-list/crl.pem

[ ssl_server ]
basicConstraints          = CA:FALSE
nsCertType                = server
keyUsage                  = digitalSignature, keyEncipherment
extendedKeyUsage          = serverAuth, nsSGC, msSGC
nsComment                 = "OpenSSL Certificate for SSL Web Server"

[ ssl_client ]
basicConstraints          = CA:FALSE
nsCertType                = client
keyUsage                  = digitalSignature, keyEncipherment
extendedKeyUsage          = clientAuth
nsComment                 = "OpenSSL Certificate for SSL Client"

[ v3_req ]
basicConstraints         = CA:FALSE
keyUsage                 = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]
basicConstraints          = critical, CA:true, pathlen:0
nsCertType                = sslCA
keyUsage                  = cRLSign, keyCertSign
extendedKeyUsage          = serverAuth, clientAuth
nsComment                 = "OpenSSL CA Certificate"

[ crl_ext ]
basicConstraints          = CA:FALSE
keyUsage                  = digitalSignature, keyEncipherment
nsComment                 = "OpenSSL generated CRL"
```

Create Certification Authority (CA)

Create folder structure to hold certificates:

```
$ mkdir /opt/met-ca

$ mkdir /opt/met-ca/certs

$ mkdir /opt/met-ca/crl

$ mkdir /opt/met-ca/newcerts

$ mkdir /opt/met-ca/private

$ mkdir /opt/met-ca/requests

$ touch /opt/met-ca/index.txt

$ echo “01” > /opt/ca/serial

$ echo “01” > /opt/ca/crlnumber

$ chmod 700 /opt/ca
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create public and private auto signed keys of our own Certification Authority (in the example is valid for 10 years). We will need to specify certificate data, password to crypt and the process will generate two files:

```
$ openssl req -config /etc/ssl/openssl.cnf -new -x509 -days 3650 -sha1 -newkey rsa:2048 -keyout /opt/ca/private/ca.key -out /opt/ca/ca.crt
```

Change key grants:

```
$ chmod 600 /opt/met-ca/private/ca.key
```

Create Apache Certification

Create a pair of keys. We will need to specify password and certification data. Will be created the file key (server.key) and certification request (server.pem).

```
$ openssl req -new -sha1 -newkey rsa:2048 -nodes -keyout server.key -out server.pem
```

CA should sign the certification request. Copy server.pem to certification request and sign.

```
$ openssl ca -config /etc/ssl/openssl.cnf -policy policy_anything -extensions ssl_server -out requests/server-signed.pem -infiles requests/server.pem
```

Apache will be waiting a PEM format, is mandatory convert to this format:

```
$ openssl x509 -in requests/server-signed.pem -out requests/server.crt
```

Apache configuration

Copy server.crt and server.key into /etc/apache2/ssl (if SSL folder not exists, create it). Change files grants (server.key should only be read by root user, server.crt should have read grant to everybody):

```
$ chmod 400 server.key
$ chmod 444 server.crt
```

If Apache is not listening SSL port, enable it configuring /etc/apache2/ports.conf:

```
Listen 80
Listen 443
```

Enable Apache SSL support:

```
$ apt-get install libapache-mod-ssl
```

Indicate Apache to load SSL modules:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
$ a2enmod ssl
```

Configure website:

Link default-ssl file to enable SSL:

```
$ ln -s /etc/apache2/sites-available/default-ssl /etc/apache2/sites-enabled/default-ssl
```

Edit /etc/apache2/sites-availables/default-ssl (enable SSL engine and set SSLCertificateFile and SSLCertificateKeyFile)

```
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

Create user certificate:

To create certificates and sign with CA, first create the pair of keys. We will need to specify password and certification data.

Will be created a eky (user.key) and certification request (user.pem):

```
$ openssl req -new -sha1 -newkey rsa:2048 -nodes -keyout user.key -out user.pem
```

With CA is needed sign certificates:

```
$ cp /tmp/usuario.pem /opt/ca/requests
$ openssl ca -config /etc/ssl/openssl.cnf -policy policy_anything -extensions [b]ssl_client[/b] -out requests/user-signed.pem -infiles requests/user.pem
```

Convert certificate to PKCS12 to be recognized by browser

```
$ openssl pkcs12 -export -clcerts -in user-signed.pem -inkey usuario.key -out user.p12
```

The files user-signed.pem, user.pem and user.key can be deleted

```
$ wipe user-signed.pem user.key user.pem
```

*The **user.p12** should be installed to allow user browser.*

Certification revocation

If the server has been compromised the CA should be revocated. To revocate a certificate should see serial number at /opt/ca/index.txt.

```
$ openssl ca -config /etc/ssl/openssl.cnf -revoke newcerts/<serial-number>.pem
```

Creat a list of revocated certificates (CRL)

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
$ openssl ca -config /etc/ssl/openssl.cnf -gencrl -crlexts crl_ext -md sha1 -out crl.pem
```

To send to the browser list should be converted to DER format:

```
$ openssl crl -in crl.pem -out ca.crl -outform DER
```

Now can be sent to browser.

Add CRL to Apache

Copy CRL file to Apache SSL folder:

```
$ cp /tmp/crl.pem /etc/apache2/ssl/ca.crl
```

Modify /etc/apache2/sites-available/default-ssl file, add SSLCARevocationFile:

```
SSLCARevocationFile /etc/apache2/ssl/ca.crl
```

Restart Apache service.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring SWFTools

By default the application comes with tomcat bundle with pre-configured SWFTools into.

In most cases it is not necessary installi SWFTools in your server. Only in case it presents some problem with pre-configured SWFTools into tomcat bundle.

Application take advantage of SWFTools to convert PDF document to SWF to be shown into UI preview tab.

See FlexPaper: Converting Documents for more info.

Ubuntu

```
sudo add-apt-repository ppa:guilhem-fr/swftools
sudo apt-get update
sudo apt-get install swftools
```

Improving performance

For better performance, use this configuration.

Property	Description
system.swftools.pdf2swf	*Pdf2swf tool used to convert PDF documents to SWF.* `/usr/bin/pdf2swf -T 9 -f -t -G -s storeallcharacters ${fileIn} -o ${fileOut}`

The parameters specifies that:

- *-f : Fonts should be embedded, improves searchability in the document*
- *-t : Inserts a stop between each frame, improves stability*
- *-G : Makes the document smaller and faster to render*
- *-s storeallcharacters : Stores all character information about the texts in the document, improves searchability*

Red Hat and CentOS

Better if you compile version 0.9.x

```
yum install zlib-devel libjpeg-devel giflib-devel freetype-devel gcc gcc-c++
wget http://www.swftools.org/swftools-0.9.1.tar.gz
tar xzf swftools-0.9.1.tar.gz
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
cd swftools-0.9.1
./configure --prefix=/usr/local
make
make install
```

Improving performance

For better performance, use this configuration.

Property	Description
system.swftools.pdf2swf	*Pdf2swf tool used to convert PDF documents to SWF.* `/usr/bin/pdf2swf -T 9 -f -t -G -s storeallcharacters ${fileIn} -o ${fileOut}`

The parameters specifies that:

- *-f : Fonts should be embedded, improves searchability in the document*
- *-t : Inserts a stop between each frame, improves stability*
- *-G : Makes the document smaller and faster to render*
- *-s storeallcharacters : Stores all character information about the texts in the document, improves searchability*

Take a look at pdf2swf wiki for complete parameters reference.

Debian

This package is not include in the Debian repositories, so you need to compile from source:

```
aptitude install build-essential libgif-dev xpdf libfreetype6 libfreetype6-dev libjpeg62 libjpeg8 libjpeg8-dev
wget http://www.swftools.org/swftools-0.9.1.tar.gz
tar xzf swftools-0.9.1.tar.gz
cd swftools-0.9.1
./configure --prefix=/usr/local
make
make install
```

Or you can try to install the Ubuntu package.

Take a look at pdf2swf wiki for complete parameters reference.

Improving performance

For better performance, use this configuration.

Property	Description
system.swftools.pdf2swf	*Pdf2swf tool used to convert PDF documents to SWF.* `/usr/bin/pdf2swf -T 9 -f -t -G -s storeallcharacters ${fileIn} -o ${fileOut}`

The parameters specifies that:

- *-f : Fonts should be embedded, improves searchability in the document*
- *-t : Inserts a stop between each frame, improves stability*
- *-G : Makes the document smaller and faster to render*
- *-s storeallcharacters : Stores all character information about the texts in the document, improves searchability*

Windows

Download last Swftools http://www.swftools.org/download.html and install it (for example at c:\tomcat-7.0.61\tools\swftools).

Improving performance

For better performance, use this configuration.

Property	Description
system.swftools.pdf2swf	*Pdf2swf tool used to convert PDF documents to swf.* `c:\tomcat-7.0.61\tools\swftools\pdf2swf.exe -T 9 -f -t -G -s storeallcharacters ${fileIn} -o ${fileOut}`

The parameters specifies that:

- *-f : Fonts should be embedded, improves searchability in the document*
- *-t : Inserts a stop between each frame, improves stability*
- *-G : Makes the document smaller and faster to render*
- *-s storeallcharacters : Stores all character information about the texts in the document, improves searchability*

Take a look at pdf2swf wiki for complete parameters reference.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Troubleshooting

Conversion problems with big files

If the document is large (has many pages) or is complex, the conversion may fail with this error:

> *ERROR ID Table overflow*
>
> *ERROR This file is too complex to render- SWF only supports 65536 shapes at once*

You can workaround this error adding the ***-s poly2bitmap*** *parameter which convert graphics to bitmaps:*

Property	*Description*
system.swftools.pdf2swf	`/usr/bin/pdf2swf -T 9 -f -t -G -s poly2bitmap -s storeallcharacters ${fileIn} -o ${fileOut}`

Segmentation fault (core dumped)

This is a log of segmentation fault error:

```
/home/openkm/tomcat-7.0.61/bin/pdf2swf -f -T 9 -t -s storeallcharacters input.pdf -o output.swf
NOTICE  processing PDF page 1 (878x1105:0:0) (move:0:0)
NOTICE  File contains forms
NOTICE  File contains pbm pictures
NOTICE  File contains soft masks
WARNING multiply blended transparency groups not yet supported!
NOTICE  File contains transparency groups
Segmentation fault (core dumped)
```

You can try the ***-s poly2bitmap*** *or* ***-s bitmap arguments****. In our experience it seems the bitmap option is more reliable for fixing display issues, but of course you decimate any text, and in some cases increase file size and processing time. You can also add the* ***-s multiply=4*** *the bitmap argument to get a really crisp bitmap output further increasing size and processing time, but gets you pretty close to the original PDF.*

```
/home/openkm/tomcat-7.0.61/bin/pdf2swf -f -T 9 -t -s bitmap -s storeallcharacters input.pdf -o output.swf
NOTICE  processing PDF page 1 (878x1105:0:0) (move:0:0)
NOTICE  Writing SWF file output.swf
```

Error with PDF with copy protected

Perhaps the PDF is copy protected. Hence, you're not allowed to convert it.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FATAL PDF disallows copying

If you created this PDF using Acrobat Distiller, try switching the copy protection checkbox off. If you can't disable the protection you can modify the 'pdf2swf source code to disable this check (see http://www.foolabs.com/xpdf/cracking.html).

The pre-configured pdf2swf tool into tomcat bundle comes with this hacking.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring mail

The application sometimes sends emails to users, for example when the users have been subscribed to a document or folder.

By default tomcat mail is configured assuming your server has a mail service binding in your localhost.

This is the default value:

```
spring.mail.host=localhost
spring.mail.port=25
spring.mail.username=none
spring.mail.password=none
spring.mail.from=noreply@openkm.com
spring.mail.properties.mail.smtp.auth=false
spring.mail.properties.mail.smtp.starttls.enable=false
```

Change mail configuration

Get your mail account data (same than you will use to configure standard mail client, like Outlook or Thunderbird).

Edit the file $TOMCAT_HOME/openkm.properties:

```
vim /home/openkm/tomcat-8.5.24/openkm.properties
```

After editing the server.xml should looks something like this:

```
spring.mail.host=localhost
spring.mail.port=25
spring.mail.username=none
spring.mail.password=none
spring.mail.from=noreply@openkm.com
spring.mail.properties.mail.smtp.auth=false
spring.mail.properties.mail.smtp.starttls.enable=false
```

Gmail example

```
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=<username>
spring.mail.password=<password>
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com
spring.mail.properties.mail.debug=true
```

Office 365

```
spring.mail.host=smtp.office365.com
spring.mail.username=<username>
spring.mail.password=<password>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.ssl.trust=smtp.office365.com
spring.mail.properties.mail.debug=true
```

Email server with old SSL configuration

```
spring.mail.host=mail.myserver.com
spring.mail.port=465
spring.mail.username=<username>
spring.mail.password=<password>
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.ssl.enable=true
spring.mail.properties.mail.debug=true
```

Additional information

Optionally, you can use these attributes :

- ***spring.mail.user="<username>"***
- ***spring.mail.test-connection=true (when enabled if the mail server connections test fails openkm will not start)***

If you get problems during tomcat startup, can try using the attribute ***spring.mail.smtp.quitwait="false"****.*

A complete list of all available parameters can be found at sun.mail.smtp package Javadoc and javax.mail package Javadoc.

Note that depending on the SMTP server configuration only the address configured in ***spring.mail.smtp.user*** *may be able to send messages, this mean, the user who appears in the "From" mail header.*

Testing configuration

Sometimes configuring mail can be complex and error-prone because there is a lot of configuration parameters and many differents mail servers. To ease this configuration we have developed a utility called OKMMailChecker which helps you to test the mail configuration whithout the need of restarting OpenKM, which takes time and make it unavailable for users during these tests.

Usage

This utility can be downloaded from https://download.openkm.com and should be placed into OpenKM directory, where the ***openkm.properties*** *is located. Indeed, this utility will read the mail configuration from this file. These two configuration properties are required, and will be used to send the test mail:*

- ***spring.mail.check.from****: The test mail origin address.*
- ***spring.mail.check.to****: The test mail destination address.*

Once these properties are set, execute the utility this way:

```
$ java -jar OKMMailChecker.jar
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat memory utilization

It's also a good idea to configure the Tomcat memory utilization.

*Edit the file $TOMCAT_HOME/bin/setenv.sh and edit the parameter **JAVA_OPTS** where you can increase the system memory managed by the JVM (Java Virtual Machine).*

OpenJDK 1.8

This is the default configuration file values. Tomcat will you 2048MB (Xmx parameter) and the size of the heap from version 1.8 is dinamically assigned by JVM. For this configuration your server needs at least >=3.5GB of RAM (Is a good practice always getting 1GB of free RAM for OS).

```
JAVA_OPTS="$JAVA_OPTS -Xms256m -Xmx2048m -XX:+UseConcMarkSweepGC"
JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true -Dlog4j.configuration=file://$CATALINA
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dmail.mime.igno
CATALINA_PID=$CATALINA_HOME/catalina.pid
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/lib/sigar
```

OpenJDK 1.7

This is the default configuration file values. Tomcat will you 2048MB (Xmx parameter) and a maximum heap of 512MB (-XX:MaxPermSize parameter). For this configuration your server needs at least >=3.5GB of RAM (Is a good practice always getting 1GB of free RAM for OS).

```
JAVA_OPTS="$JAVA_OPTS -Xms256m -Xmx2048m -XX:PermSize=256m -XX:MaxPermSize=512m -XX:+
JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true -Dlog4j.configuration=file://$CATALINA
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dmail.mime.igno
CATALINA_PID=$CATALINA_HOME/catalina.pid
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/lib/sigar
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat port

By default the tomcat binding ports are 8005, 8080 and 8009.

If you have other tomcat running on same server or other application like JBoss Application Server, probably these ports are already used. In this case you should change default ports.

Change tomcat ports

The port change must be done with the tomcat application stopped.

Edit $TOMCAT_HOME/confg/server.xml and look for all Tomcat port properties.

Service	Default port	Example of new port
Shut-down Port	8005	8105

	Change
	`<Server port="8005" shutdown="SHUTDOWN">`
	to
	`<Server port="8105" shutdown="SHUTDOWN">`	

Tomcat Connector Port	8080	8180

	Change
	`<Connector address="0.0.0.0" connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>`
	to
	`<Connector address="0.0.0.0" connectionTimeout="20000" port="8180" protocol="HTTP/1.1" redirectPort="8443"/>`	

AJP Connector Port	8009	8109
	Change	

```
<Connector address="127.0.0.1" port="8009" protocol="AJP/1.3"
redirectPort="8443"/>
```

to

```
<Connector address="127.0.0.1" port="8109" protocol="AJP/1.3"
redirectPort="8443"/>
```

*To shut-down the different instances execute the **shutdown.sh** script from each $TOMCAT_HOME/bin.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat session time-out

By default the session time-out is set to 30 minutes.

To change the Tomcat session time-out, edit the $TOMCAT_HOME/conf/web.xml and change the values into the session section:

```
<!-- You can set the default session timeout (in minutes) for all newly   -->
<!-- created sessions by modifying the value below.                       -->
<session-config>
    <session-timeout>30</session-timeout>
</session-config>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat SSL/TLS

Secure Sockets Layer (SSL) and the new Transport Layer Security (TLS), are cryptographic protocols designed to provide a secure communication between two computers. They use certificates and cryptography to authenticate the counterpart with whom they are communicating and to negotiate a session key. This session key is used to encrypt data between the endpoints. A consequence of using certificates, is that Certificate Authorities (CA) and a public key infrastructure are necessary to verify the relation between a certificate and its owner, as well as to generate, sign, and administer the validity of certificates. Due to this reason you have two options to configure SSL:

- ***Self-signed Certificate**: it's free because you create your own certificate.*
- ***CA Certificate**: It cost money because you need to pay for it to a certification authority.*

Tomcat currently operates only on JKS, PKCS11 or PKCS12 format keystores. The JKS format is Java's standard "Java KeyStore" format, and is the format created by the key tool command-line utility. This tool is included in the OpenJDK. The PKCS12 format is an Internet standard, and can be manipulated via (among other things) OpenSSL and Microsoft's Key-Manager.

Each entry in a key store is identified by an alias string. Whilst many key store implementations treat aliases in a case non sensitive manner, case sensitive implementations are available. The PKCS11 specification, for example, requires that aliases are case sensitive. To avoid issues related to the case sensitivity of aliases, it is not recommended to use aliases that differ only in case.

Self-signed Certificate

Create the Certificate Key store

```
$ keytool -genkey -alias tomcat -keyalg RSA -keystore self-signed.jks
Enter keystore password: ********
Re-enter new password: ********
What is your first and last name? [Unknown]: Self-Signed Certificate
What is the name of your organizational unit? [Unknown]: System Management
What is the name of your organization? [Unknown]: OpenKM
What is the name of your City or Locality? [Unknown]: Palma de Mallorca
What is the name of your State or Province? [Unknown]: Islas Baleares
What is the two-letter country code for this unit? [Unknown]: ES
Is CN=Self-Signed Certificate, OU=System Management, O=OpenKM, L=Palma de Mallorca, ST=Islas Baleares, C=ES
correct?
[no]: yes
```

Enter key password for <tomcat>

(RETURN if same as keystore password):

*The password **changeit** is the default password set into $TOMCAT_HOME/conf/server.xml file*

Edit the Tomcat Configuration file

Enable the SSL connector:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="${catalina.home}/sef-signed.jks" keystorePass="changeit" key
```

Check configuration

*Restart Tomcat service and check your OpenKM URL with **https://YOUR_SERVER_IP:8443/OpenKM**.*

CA Certificate

Prepare the Certificate Key store

$ keytool -genkey -alias tomcat -keyalg RSA -keystore ca-signed.jks

*Enter keystore password: *********

*Re-enter new password: *********

What is your first and last name? [Unknown]: CA-Signed Certificate

What is the name of your organizational unit? [Unknown]: System Management

What is the name of your organization? [Unknown]: OpenKM

What is the name of your City or Locality? [Unknown]: Palma de Mallorca

What is the name of your State or Province? [Unknown]: Islas Baleares

What is the two-letter country code for this unit? [Unknown]: ES

Is CN=Self-Signed Certificate, OU=System Management, O=OpenKM, L=Palma de Mallorca, ST=Islas Baleares, C=ES

correct?

[no]: yes

Enter key password for <tomcat>

(RETURN if same as keystore password):

*The password **changeit** is the default password set into $TOMCAT_HOME/conf/server.xml file*

Create the Certificate Signing Request

*This step is different from the self-signed certificate because you need to send this information to the Certification Authority (CA) which need to validate and sign. Your chosen CA will send you back some files with the Root Certificate the Intermediate Certificates and the Primary Certificate (which is the certificate created for you). This way we generate the key in the keystore and also generate the certificate signing request file (**ca-signed.csr**):*

```
$ keytool -certreq -keyalg RSA -alias tomcat -file ca-signed.csr -keystore ca-signed.jks
```

Once the Certification Authority reviews and validates the info, he will send you:

- *The domain certificate you have purchased*
- *The intermediate certificate*
- *The root certificate*

Or maybe they send only a file with both the intermediate certificate and the root certificate.

Install the Root Certificate

Every time you install a certificate to the key store you must enter the key store password that you chose when you generated it. Enter the following command to install the Root Certificate file:

```
$ keytool -import -trustcacerts -alias root -file RootCertFileName.crt -keystore ca-signed.jks
```

If you receive a message that says "Certificate already exists in system-wide CA keystore under alias <...> Do you still want to add it to your own keystore? [no]:", select Yes. If successful, you will see "Certificate was added to keystore".

Install the Intermediate Certificate

If your certificate authority provided an intermediate certificate file, you will need to install it here by typing the following command:

```
$ keytool -import -trustcacerts -alias intermediate -file IntermediateCertFileName.crt -keystore ca-signed.jks
```

If successful, you will see "Certificate was added to keystore".

Install the Primary Certificate

Type the following command to install the Primary certificate file (for your domain name):

```
$ keytool -import -trustcacerts -alias tomcat -file PrimaryCertFileName.crt -keystore ca-signed.jks
```

The command $ keytool -list -keystore ca-signed.jks -v shows all certificates imported.

Edit the Tomcat Configuration file

Enable the SSL connector:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="${catalina.home}/ca-signed.jks" keystorePass="changeit" keyA
```

Check configuration

Restart Tomcat service and check your OpenKM URL with ***https://YOUR_SERVER_IP:8443/OpenKM****.*

CA Certificate without preparing keystore

Sometimes the person who installs the certificate is not the same that made the request to obtain the certificate. In this case every step is similar to the previous one with the exception of the installation of the Primary Certificate.

In this case, when the primary certificate is installed, it is necesary to install the private RSA key too. To do it, apart from the primary certificate, the private RSA key must be given.

Obtain p12 certficate

Obtain p12 certificate from private key and primary certificate. This is the way that both keys are installed.

```
$ openssl pkcs12 -export -in PrimaryCertFileName.crt -inkey privateRSA.key > PrimaryCertFileName.p12
```

If successful, PrimaryCertFileName.p12 certificate will be created.

Install p12 Certificate

Type the following command to install the .p12 certificate file (for your domain name):

```
$ keytool -v -importkeystore -srckeystore PrimaryCertFileName.p12 -srcstoretype PKCS12 -destkeystore ca-signed.jks -
deststoretype JKS
```

If successful, you will see "Certificate was added to keystore".

Change alias to Primary Certificate

Type the following command to install the .p12 certificate file (for your domain name):

```
$ keytool -changealias -keystore ca-signed.jks -alias 1 -destalias tomcat
```

Let's Encrypt

Let's Encrypt uses a client application which automates the certificate generation and renovation.

Client installation

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

In Linux you can install the tool with these commands:

```
$ wget https://dl.eff.org/certbot-auto
$ chmod a+x certbot-auto
```

The first will install the required dependencies:

```
$ sudo ./certbot-auto
```

Certificate generation

Now it's time to generate a certificate. This is a sample command line used to generate only the certificate. For more info about usage and command line options, please visit https://certbot.eff.org/:

```
$ sudo ./certbot-auto certonly --standalone -m contact@mail-server.com -d sample-domain.com
```

This will create a directory structure where you can locate the certificate. This directory is protected so you need to switch to root to go into:

```
$ sudo su -
$ cd /etc/letsencrypt/live/sample-domain.com
```

As you can see there are seveal PEM files. Let's create a PKCS12 certificate from these files which will be used in Tomcat:

```
$ openssl pkcs12 -export -in cert.pem -inkey privkey.pem -CAfile chain.pem -caname root -name tomcat -out mycert.p12
```

Keep the password safe because it will be used later.

Tomcat configuration

Copy the generated file to TOMCAT_HOME and edit the $TOMCAT_HOME/conf/server.xml file to configure the new certificate:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="${catalina.home}/mycert.p12" keystorePass="changeit" keyAlia
```

Once Tomcat is started, the 8443 port will accept SSL connections. But remember these connection should only be done using the domain you used to generate the certificate, otherwise the browser will alert you about the site.

Other configuration

To redirect from non SSL port 8080 to SSL port 8443 edit the $TOMCAT_HOME/web.xml and append at the end:

```
<security-constraint>
```

```
    <web-resource-collection>
        <web-resource-name>App_nmae</web-resource-name>
        <url-pattern>/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
    </web-resource-collection>

    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

And restart Tomcat.

Known issues

Windows uses a different protocol

In Window systems the protocol used in the Connector must be changed so the final configuration is:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol" SSLEnabl
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="${catalina.home}\ca-signed.jks" keystorePass="changeit" keyA
```

ca-bundles certificates

Sometimes the certificates provider gives serveral certifcates in one file with .ca-bundle extension. Sometimes, when this bundle is imported as one file, only one of the certicates is imported. In is this case, the solution is to split the bundle in several files (once for each certificate).

To do it, it is possible to edit the ca-bundle file, and each certificate is separated by:

```
-----BEGIN CERTIFICATE-----
        ....
        ....
-----END CERTIFICATE-----
```

Additional information

Are several ways to configure SSL in Tomcat for more information:

- *Apache Tomcat - SSL / TLS Configuration HOW-TO*
- *A Simple, Step-By-Step Guide To Apache Tomcat SSL Configuration | MuleSoft*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat timezone

Tomcat works in GMT timezone if you want to change it you should add something like at $TOMCAT_HOME/bin/setenv.sh (or $TOMCAT_HOME/bin/setenv.bat if you're on Windows):

```
CATALINA_OPTS="-Duser.timezone=Asia/Hong_Kong"
```

More information about timezone list.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring openkm.xml

By default OpenKM is not using openkm.xml from the file system, it is using the openkm.xml file what is into openkm.war

By default OpenKM comes with tree authentication providers:

- *Superuser in memory.*
- *Database.*
- *Active Directory (LDAP).*

The configuration added into ***openkm.xml*** *file is* ***loaded only*** *if all the OpenKM* ***internal security providers are disabled***.

You should disable internal authentication in the openkm.properties:

```
#Authentication
okm.authentication.database=false
okm.authentication.supervisor=false
okm.authentication.ldap=false
```

Default openkm.xml configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin

</beans:beans>
```

Enable external openkm.xml

To enable external openkm.xml file you must change the configuration parameter named ***okm.authentication.config*** *in the* ***openkm.properties*** *file. The values of the parameter* ***okm.authentication.config*** *might be the file system path to the openkm.xml file.*

Example:

```
okm.authentication.config=file:/home/openkm/tomcat-8.5.4/openkm.xml
```

Memory sample

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin

  <!-- Security delegated to classes -->
  <security:authentication-manager alias="authenticationManager">
 <security:authentication-provider>
  <security:user-service>
            <security:user name="okmAdmin" password="admin" authorities="ROLE_ADMIN"
        </security:user-service>
    </security:authentication-provider>
  </security:authentication-manager>

</beans:beans>
```

Database authentication sample

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns:amq="http://activemq.apache.org/schema/core"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta
                                 http://activemq.apache.org/schema/core
                                 http://activemq.apache.org/schema/core/activemq-core

  <!-- Security delegated to classes -->
  <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider>
      <security:password-encoder hash="bcrypt"/>
      <security:jdbc-user-service
        data-source-ref="dataSource"
        users-by-username-query="select usr_id, usr_password, 1 from OKM_USER where u
        authorities-by-username-query="select ur_user, ur_role from OKM_USER_ROLE whe
    </security:authentication-provider>
  </security:authentication-manager>

</beans:beans>
```

Ldap Authentication sample

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin

  <!-- Security delegated to classes -->
```

```
  <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

 <beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSprin
  <beans:constructor-arg value="ldap://192.168.1.40"/>
  <beans:property name="userDn" value="CN=Administrator,CN=Users,DC=openkm,DC=local"/
  <beans:property name="password" value="Darkman2017i"/>
  <beans:property name="baseEnvironmentProperties">
    <beans:map>
     <beans:entry>
       <beans:key>
       <beans:value>java.naming.referral</beans:value>
      </beans:key>
      <beans:value>follow</beans:value>
     </beans:entry>
    </beans:map>
   </beans:property>
 </beans:bean>

 <beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentic
  <beans:constructor-arg>
   <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentica
    <beans:constructor-arg ref="contextSource"/>
    <beans:property name="userSearch" ref="userSearch"/>
   </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
 </beans:bean>

 <beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterBa
  <beans:constructor-arg index="0" value="DC=openkm,DC=local" />
  <beans:constructor-arg index="1" value="(sAMAccountName={0})" />
  <beans:constructor-arg index="2" ref="contextSource" />
  <beans:property name="searchSubtree" value="true" />
 </beans:bean>

 <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.security
  <beans:constructor-arg ref="contextSource"/>
  <beans:constructor-arg value="DC=openkm,DC=local"/>
  <beans:property name="groupSearchFilter" value="member={0}"/>
  <beans:property name="groupRoleAttribute" value="cn"/>
  <beans:property name="searchSubtree" value="true" />
  <beans:property name="convertToUpperCase" value="false" />
  <beans:property name="rolePrefix" value="" />
 </beans:bean>

 <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
  <beans:constructor-arg ref="userSearch"/>
  <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
 </beans:bean>

</beans:beans>
```

Multiple providers sample

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns:amq="http://activemq.apache.org/schema/core"
```

```
            xsi:schemaLocation="http://www.springframework.org/schema/beans
                                http://www.springframework.org/schema/beans/spring-b
                                http://www.springframework.org/schema/security
                                http://www.springframework.org/schema/security/sprin
                                http://www.springframework.org/schema/task
                                http://www.springframework.org/schema/task/spring-ta
                                http://activemq.apache.org/schema/core
                                http://activemq.apache.org/schema/core/activemq-core

  <!-- Security delegated to classes -->
  <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider>
   <security:user-service>
        <security:user name="okmAdmin" password="admin" authorities="ROLE_ADMIN" />
      </security:user-service>
    </security:authentication-provider>
    <security:authentication-provider>
      <security:password-encoder hash="bcrypt"/>
      <security:jdbc-user-service
        data-source-ref="dataSource"
        users-by-username-query="select usr_id, usr_password, 1 from OKM_USER where u
        authorities-by-username-query="select ur_user, ur_role from OKM_USER_ROLE whe
    </security:authentication-provider>
  </security:authentication-manager>

</beans:beans>
```

Configuring jPBM

In OpenKM v7.1 the workflow engine is deployed as an independient application to ensure security and stability.

Please, take a look at JBPM Console for more info.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring safe trash purge

When an user delete a node (document, folder, mail or record) from OpenKM, it goes to user trash. Periodically users should empty their trash to free repository space.

When the user empties the trash, the documents are permanently lost.

That means if accidentally the trash has been cleaned, the nodes only can be restored from a previous backup. Usually restoring a full database and repository backup, which is a tedious task.

Administrator can clean all application trashes with Purge trashes utility.

You can configure a place where the documents will go after being purged from trash.

If you need to recover accidentally deleted node, then will not be necessary restore the entire application from backup.

*You can configure a directory in the server file system where an export of the trash to be placed, using the "**repository.purgatory.home**" configuration parameter. Each time before a node is definitely removed from trash, the application does an export (like Repository export feature) to the purgatory home folder, creating into sub folder structure "**YEAR/MONTH/DAY**" what can help the administrator to locate the data to be recovered. To recover the data use the Repository import feature with the **Metadata** option checked.*

Field / Property	*Type*	*Description*
repository.purgatory.home	***String***	*Set the safe trash purge destination folder.* `/mnt/backup/trash`

Is a good practice to backup the purgatory home folder, at least for some prudential time (1-2 months depending on your company policy).

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Tomcat to work behing an Internet proxy

We encourage to consider do not configure Internet connection in Servers behing an Internet proxy.

It have some disadvantages like:

- *You will need to use your proxy data each time you will upgrade the application with OpenKM automatic updater tools.*
- *It's need to configure JVM parameters with your Internet proxy data for the application be able to connect to Internet.*

Is not much clear what are the advantatges for this kind of configuration on Production Servers, what are not the same scenario than end users. ***Take it as is, only a suggestion****.*

Linux

Configuration below is based on setting proxy configuration parameters in the JVM environment. There's another option what is configuring proxy at OS level, here are several option one of them is:

*Edit your "****/etc/bash.bashrc****" file as root and put these lines at the end of your "****/etc/bash.bashrc****" file:*

```
export http_proxy=http://username:password@proxyserver.net:port/
export ftp_proxy=http://username:password@proxyserver.net:port/
```

- ***Stop OpenKM****.*
- ***Edit the $TOMCAT_HOME/bin/setenv.sh*** *file and add the lines:*

The http://proxy.url.com value must be your own url host of the proxy.

*The "****user****" value must be the user for getting connection to the proxy. If your* ***proxy does not require authentication you can remove the entire line*** *where are set these parameters.*

*The "****password****" values must be the password of the user for getting connection to the proxy. If your* ***proxy does not require authentication you can remove the entire line*** *where are set these parameters.*

```
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyHost=http://proxy.url.com -Dhttps.proxyHost=htt
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyUser=user -Dhttps.proxyUser=user
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyPassword=password -Dhttps.proxyPassword=passwor
```

At the end ***your setenv.sh should looking something like that*** *:*

```
JAVA_OPTS="$JAVA_OPTS -Xms256m -Xmx2048m -XX:+UseConcMarkSweepGC -Djava.awt.headless=
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dmail.mime.igno
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyHost=http://proxy.url.com -Dhttps.proxyHost=htt
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyUser=user -Dhttps.proxyUser=user
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyPassword=password -Dhttps.proxyPassword=passwor
CATALINA_PID=$CATALINA_HOME/catalina.pid
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/lib/sigar
```

If you have issues resolving DNS you can also try with this configuration what is trying to use Google DNS:

```
set JAVA_OPTS=%JAVA_OPTS% -Dsun.net.spi.nameservice.nameservers=8.8.8.8
set JAVA_OPTS=%JAVA_OPTS% -Dsun.net.spi.nameservice.provider.1=dns,sun
```

- ***Start OpenKM***.
- *Go to **Administration** > **Tools** > Scripting and execute the script below (should be shown google html code).*

```
import java.net.URL;
import java.net.HttpURLConnection;
import java.io.BufferedReader;

URL url = new URL("http://www.google.com");
HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
urlConn.setDoInput(true);
urlConn.setDoOutput(true);
urlConn.setUseCaches(false);
urlConn.setRequestMethod("GET");
BufferedReader input = new BufferedReader(new InputStreamReader(urlConn.getInputStrea
String line;
StringBuilder sb = new StringBuilder();
while ((line = input.readLine()) != null) {
 sb.append(line);
}
input.close();
print(sb.toString().replaceAll("\n","<br/>"));
```

Windows

Configuration below is based on setting proxy configuration parameters in the JVM environment. There's another option what is configuring proxy at OS level, consider looking for this information at Microsoft technet website.

Step one

Check starting and stopping OpenKM from the command line.

- *First stop OpenKM service.*
- *Edit **$TOMCAT_HOME/bin/seten.bat** file and add the lines:*

The http://proxy.url.com value must be your own url host of the proxy.

*The "**user**" value must be the user for getting connection to the proxy. If your **proxy does not require authentication you can remove the entire line** where are set these parameters.*

*The "**password**" values must be the password of the user for getting connection to the proxy. If your **proxy does***

not require authentication you can remove the entire line *where are set these parameters.*

```
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyHost=http://proxy.url.com -Dhttps.proxyHost=htt
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyUser=user -Dhttps.proxyUser=user
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyPassword=password -Dhttps.proxyPassword=passwor
```

At the end ***your setenv.bat should looking something like that****:*

```
JAVA_OPTS="$JAVA_OPTS -Xms256m -Xmx2048m -XX:+UseConcMarkSweepGC -Djava.awt.headless=
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dmail.mime.igno
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyHost=http://proxy.url.com -Dhttps.proxyHost=htt
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyUser=user -Dhttps.proxyUser=user
set JAVA_OPTS=%JAVA_OPTS% -Dhttp.proxyPassword=password -Dhttps.proxyPassword=passwor
CATALINA_PID=$CATALINA_HOME/catalina.pid
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/lib/sigar
```

If you have issues resolving DNS you can also try with this configuration what is trying to use Google DNS:

```
set JAVA_OPTS=%JAVA_OPTS% -Dsun.net.spi.nameservice.nameservers=8.8.8.8
set JAVA_OPTS=%JAVA_OPTS% -Dsun.net.spi.nameservice.provider.1=dns,sun
```

- ***Start OpenKM from the command line***.

```
c:\tomcat-8.5.24\bin\> catalina.bat run
```

- *Go to* ***Administration*** *>* ***Tools*** *>* *Scripting and execute the script below (should be shown google html code).*

```
import java.net.URL;
import java.net.HttpURLConnection;
import java.io.BufferedReader;

URL url = new URL("http://www.google.com");
HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
urlConn.setDoInput(true);
urlConn.setDoOutput(true);
urlConn.setUseCaches(false);
urlConn.setRequestMethod("GET");
BufferedReader input = new BufferedReader(new InputStreamReader(urlConn.getInputStrea
String line;
StringBuilder sb = new StringBuilder();
while ((line = input.readLine()) != null) {
 sb.append(line);
}
input.close();
print(sb.toString().replaceAll("\n","<br/>"));
```

Step two

If all has gone right in step one, now you can configure parameters into the OpenKM service.

- ***Stop OpenKM***.

- *Go to **$TOMCAT_HOME/bin** and execute the next command line:*

The command lines below does not registering the parameters for DNS resolving. If you need it, simply add them.

```
c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttp.proxyHost=http://proxy.url.com"

c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttps.proxyHost=http://proxy.url.com"

c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttp.proxyUser=user"

c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttps.proxyUser=user"

c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttp.proxyPassword=password"

c:\tomcat-8.5.24\bin\> OpenKM //US//OpenKM ++JvmOptions="-Dhttps.proxyPassword=password
```

- ***Start OpenKM from services***.
- *Go to **Administration** > **Tools** > Scripting and execute the script below (should be shown google html code).*

```
import java.net.URL;
import java.net.HttpURLConnection;
import java.io.BufferedReader;

URL url = new URL("http://www.google.com");
HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
urlConn.setDoInput(true);
urlConn.setDoOutput(true);
urlConn.setUseCaches(false);
urlConn.setRequestMethod("GET");
BufferedReader input = new BufferedReader(new InputStreamReader(urlConn.getInputStrea
String line;
StringBuilder sb = new StringBuilder();
while ((line = input.readLine()) != null) {
 sb.append(line);
}
input.close();
print(sb.toString().replaceAll("\n","<br/>"));
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change repository home

*By default the repository is located at "**$TOMCAT_HOME/repository**" but you can change to other location.*

> *This action must be done with application stopped.*

Understanding the contents of repository folders

*Into "**$TOMCAT_HOME/repository**" there are three folder **tmp**, **index**, **cache** and **datastore**.*

Folder	Backup	Description
tmp	No	This folder contains some temporary files to be shared beween several clusteres OpenKM instances.
index	No	The folder which contains the Lucene search engine indexes. The content of that folder can be rebuild. *Although is not mandatory backup this folder, we recommend doing it.*
cache	No	The folder which contains the application cached files. For example, for previewing documents, sometimes is needed to transform the original document to other format. The conversion consumes a lot of hardware resources, for this reason the application stores these converted files to be reused later. The content of that folder can be cleaned without major problems. *Although is not mandatory backup this folder, we recommend doing it.*
datastore	Yes	The folder which contains all binary data of the documents. *Is **mandatory mandatory backup** this folder otherwise the binary data of the documents could be lost on a server hardware fail.*
extraction	Yes	This folder contains the text extracted from those documents stored in OpenKM. A backup is not really necessary but may be a time saver.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Move entire repository location

- *Stop the application*
- *Move the "**$TOMCAT_HOME/repository**" to other location.*
- *Edit the "**$TOMCAT_HOME/openkm.properties**" file and add the property "**repository.home**".*
- *Start the application*

Sample1

> *From OpenKM version 7.1.x and upper when the "**repository.home**" paremeter is set, the **location** in the file system **of the lucene index is not affected**. You should use the specific parameter named "**spring.jpa.properties.hibernate.search.default.indexBase**" to move the lucene index in a new location.*

Windows

```
repository.home=C:/okmrepo
```

Linux

```
repository.home=/mnt/okmrepo
```

Move repository folders individually

Table of properties:

Folder	*Property*	
index	`spring.jpa.properties.hibernate.search.default.indexBase`	`${catalina.hor`
tmp	`repository.temp.home`	`${repository.h`
cache	`repository.cache.home`	`${repository.h`
datastore	`repository.datastore.home`	`${repository.h`
extraction	`repository.extraction.home`	`${repository.h`

> *These parameters can be used in combination with the parameter "**repository.home**" and "**tenant.id**".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Stop the application*
- *Move the "**$TOMCAT_HOME/repository/{folder}**" to other location.*
- *Edit the "**$TOMCAT_HOME/openkm.properties**" file and add the some property.*
- *Start the application*

Sample

Windows

```
spring.jpa.properties.hibernate.search.default.indexBase=C:/lucene/index
repository.datastore.home=D:/okm/${repository.home}/datastore
repository.cache.home=D:/okm/${repository.home}/cache
repository.temp.home=D:/okm/${repository.home}/tmp
```

Linux

```
spring.jpa.properties.hibernate.search.default.indexBase=/mnt/lucene/index
repository.datastore.home=/mnt/okm/${repository.home}/datastore
repository.cache.home=/mnt/okm/${repository.home}/cache
repository.temp.home=/mnt/okm/${repository.home}/tmp
```

Troubleshooting

Windows mapped network disk is not working

We encourage do not use networking (NAS or Samba) for storing repository. Because network issues will affect the repository we consider as a dangerous configuration.

*Seems there're restrictions using network unit (mapped with "**net use**" command line) and application running from services.*

To solve it is necessary to create a link between UNC and the file system.

```
MKLINK /D C:\okm-repository \\192.168.1.36\eDocuments\okm-repository
```

In most cases will be necessary to use a standard user for running OpenKM service either the standard configured. Take a look at OpenKM windows service configuration to change the default "Local system user" to your own.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change default role names

*By default application comes with two connection roles named "**ROLE_USER**" and "**ROLE_ADMIN**". Any OpenKM user must have assigned one of these roles to loggin.*

Change the default roles

Change roles usually is not an easy task, we recommend have a backup before start working on it.

Is a good practice, first test this kind of changes on a development environment before trying on your production.

To change the default roles must be done three actions:

1. ***Change the roles name** at administration.*
2. ***Modify the contents of the file appContext.xml** into the "**$TOMCAT/webapp/OpenKM.war**".*
3. ***Update existing database roles** name "**ROLE_USER**" and "**ROLE_ADMIN**" to new names.*

*You can modify the file "**appContext.xml**" into the "**$TOMCAT_HOME/webapps/OpenKM.war**" or modify the exploded war file at "**$TOMCAT_HOME/webapp/OpenKM/WEB-INF/appContext.xml**".*

*Take in mind each time the "**OpenKM.war**" file is modified the contents of "**$TOMCAT_HOME/webapp/OpenKM" folder will be completely cleaned.***

Sections into the "appContext.xml" to be modified:

Mandatory:

```
<security:intercept-url pattern="/admin/**" access="ROLE_ADMIN" />
```

Optional:

```
<!-- Remove prefix to be able of use custom roles -->
<beans:bean id="roleVoter" class="org.springframework.security.access.vote.Ro
    <beans:property name="rolePrefix" value="ROLE_"/>
</beans:bean>
```

- *Go to **Administration** > Configuration parameters.*
- *Change the values of parameters "**default.admin.role**" and "**default.admin.role**". Also can be done from Database query, see the next SQL script.*
- *Go to **Administration** > **Utilities** > Database query and execute the role name update sql:*

```
-- Optionally the configuration parameters can be set from SQL
UPDATE OKM_CONFIG SET CFG_VALUE='NEW_ROLE_USER_NAME' WHERE CFG_KEY='default.user.role
UPDATE OKM_CONFIG SET CFG_VALUE='NEW_ROLE_ADMIN_NAME' WHERE CFG_KEY='default.admin.ro
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
-- Update security table
UPDATE OKM_NODE_ROLE_PERMISSION SET NRP_ROLE='NEW_ROLE_USER_NAME' WHERE NRP_ROLE='ROL
UPDATE OKM_NODE_ROLE_PERMISSION SET NRP_ROLE='NEW_ROLE_ADMIN_NAME' WHERE NRP_ROLE='RO
-- Insert new roles
INSERT INTO OKM_ROLE (ROL_ID, ROL_ACTIVE) VALUES ('NEW_ROLE_USER_NAME', 'T');
INSERT INTO OKM_ROLE (ROL_ID, ROL_ACTIVE) VALUES ('NEW_ROLE_ADMIN_NAME', 'T');
-- Update user roles
UPDATE OKM_USER_ROLE SET UR_ROLE='NEW_ROLE_USER_NAME' WHERE UR_ROLE='ROLE_USER';
UPDATE OKM_USER_ROLE SET UR_ROLE='NEW_ROLE_ADMIN_NAME' WHERE UR_ROLE='ROLE_ADMIN';
-- Remove older roles
DELETE FROM OKM_ROLE WHERE ROL_ID IN ('ROLE_USER', 'ROLE_ADMIN');
```

- *Stop the application.*
- *Modify the contents of the file "**appContext.xml**".*
- *Start the application.*

Sample of appContext.xml file

The "ROLE_USER" has been modified to "OPENKM_ROLE_USER".

The "ROLE_ADMIN" has been modified to "OPENKM_ROLE_ADMIN".

The contents of the "appContext.xml" can be modified, take only as a reference.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:context="http://www.springframework.org/schema/context"
             xmlns:jee="http://www.springframework.org/schema/jee"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/context
                                 http://www.springframework.org/schema/context/spring
                                 http://www.springframework.org/schema/jee
                                 http://www.springframework.org/schema/jee/spring-jee

    <context:component-scan base-package="com.openkm"/>

    <!-- OpenKM API -->
    <beans:import resource="soap.xml" />
    <beans:import resource="rest.xml" />
    <beans:import resource="cmis.xml" />

    <!--
    <beans:bean class="org.springframework.beans.factory.config.MethodInvokingFactory
        <beans:property name="targetClass" value="org.springframework.security.core.c
        <beans:property name="targetMethod" value="setStrategyName" />
        <beans:property name="arguments" value="_INHERITABLETHREADLOCAL" />
    </beans:bean>
    -->

    <security:global-method-security secured-annotations="enabled"/>

    <!-- Remove prefix to be able of use custom roles -->
```

```
    <beans:bean id="roleVoter" class="org.springframework.security.access.vote.RoleVo
        <beans:property name="rolePrefix" value="OPENKM_ROLE_"/>
    </beans:bean>

    <!-- Status -->
    <security:http pattern="/Status" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- Download -->
    <security:http pattern="/Download" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
     <security:custom-filter position="PRE_AUTH_FILTER" ref="downloadTokenFilter" />
    </security:http>

    <beans:bean name="downloadTokenFilter" class="com.openkm.spring.DownloadTokenFilt

    <!-- Workflow deploy -->
    <security:http pattern="/workflow-register" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- WebDAV using Basic authentication -->
    <security:http pattern="/webdav/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- Syndication using Basic authentication -->
    <security:http pattern="/feed/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- OpenCMIS (Browser) using Basic authentication -->
    <security:http pattern="/cmis/browser/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- OpenCMIS (AtomPub) using Basic authentication -->
    <security:http pattern="/cmis/atom/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- OpenCMIS (AtomPub) using Basic authentication -->
    <security:http pattern="/cmis/atom11/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- REST -->
    <security:http pattern="/services/rest/**" create-session="stateless">
        <security:intercept-url pattern="/**" access="IS_AUTHENTICATED_FULLY" />
        <security:http-basic />
    </security:http>

    <!-- Additional filter chain for normal users, matching all other requests -->
    <!-- http://info.michael-simons.eu/2011/01/28/disable-jsessionid-path-parameter-i
    <security:http access-decision-manager-ref="accessDecisionManager" access-denied-

        <!-- GWT -->
        <security:intercept-url pattern="/frontend/**" access="IS_AUTHENTICATED_FULLY
```

```
        <!-- JSPs -->
        <security:intercept-url pattern="/admin/**" access="OPENKM_ROLE_ADMIN" />
        <security:intercept-url pattern="/mobile/**" access="IS_AUTHENTICATED_FULLY"
        <security:intercept-url pattern="/custom/**" access="IS_AUTHENTICATED_ANONYMO

        <!-- Servlets -->
        <security:intercept-url pattern="/login" access="IS_AUTHENTICATED_ANONYMOUSLY
        <security:intercept-url pattern="/RepositoryStartup" access="IS_AUTHENTICATED
        <security:intercept-url pattern="/TextToSpeech" access="IS_AUTHENTICATED_FULL
        <security:intercept-url pattern="/HtmlPreview" access="IS_AUTHENTICATED_FULLY
        <security:intercept-url pattern="/SyntaxHighlighter" access="IS_AUTHENTICATED
        <security:intercept-url pattern="/WebPageImport" access="IS_AUTHENTICATED_FUL
        <security:intercept-url pattern="/Test" access="IS_AUTHENTICATED_FULLY" />

        <!-- Extensions -->
        <security:intercept-url pattern="/extension/ZohoFileUpload" access="IS_AUTHEN
        <security:intercept-url pattern="/extension/**" access="IS_AUTHENTICATED_FULL

        <!-- Login page -->
        <security:form-login login-page="/login" authentication-failure-url="/login?e

    </security:http>

    <!-- Needed for changing default role prefix -->
    <beans:bean id="accessDecisionManager" class="org.springframework.security.access
        <beans:property name="decisionVoters">
          <beans:list>
             <beans:bean class="org.springframework.security.web.access.expression.We
             <beans:ref bean="roleVoter" />
             <beans:bean class="org.springframework.security.access.vote.Authenticate
          </beans:list>
       </beans:property>
    </beans:bean>

    <!-- Security access logger -->
    <beans:bean id="loggerListener" class="com.openkm.spring.LoggerListener" />

    <jee:jndi-lookup id="dataSource" jndi-name="jdbc/OpenKMDS" resource-ref="true" />

    <!-- Security configuration moved to $CATALINA_HOME/openkm.xml -->
    <!-- WINFIX
    <security:authentication-manager alias="authenticationManager">
        <security:authentication-provider>
            <security:password-encoder hash="md5"/>
            <security:jdbc-user-service
                data-source-ref="dataSource"
                users-by-username-query="select usr_id, usr_password, 1 from OKM_USER
                authorities-by-username-query="select ur_user, ur_role from OKM_USER_
        </security:authentication-provider>
    </security:authentication-manager>
    WINFIX -->
</beans:beans>
```

Tools

Download the changeRolesTools.zip.

It includes:

- *change_roles.sh script to modify the appContext.xml file.*
- *path folder to patch "**OpenKM.war**" file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*To patch "**OpenKM.war**" file, copy the file into patch folder and then execute the patch.sh script (or patch.bat file on Windows).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Enable extensions

After the extensions have been registered in OpenKM will not be visible for the users until you enable them from the Profile General tab .

By default OpenKM comes with the extensions disabled.

Types of extensions:

- *Do not require additional software configuration.*
- *Need additional software configuration.*
- *Sample extensions for development.*

Available extensions are registered in the OKM_EXTENSION table. To enable any of them, it's only necessary to insert it in this table.

Extensions that do not require additional software configuration:

- *Go to **Administration > Utilities > Database query**. At bottom right select option **JDBC**, **paste into the box the SQL inserts** of the extensions you want to enable. Finally click at bottom right the button **execute**.*

*The **Stapling** extension is deprecated and has been superseded by **Relation**.*

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('54743940-0761-11e0-81e0-08002
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('93faba30-0c34-11e0-81e0-08002
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('4ef6ad10-1a5b-11e0-ac64-08002
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('88ca0d10-39e2-11e0-9207-08002
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('522e7720-5c54-11e0-80e3-08002
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('fa7f4556-3249-4268-88e0-0dd78
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('808e7a42-2e73-470c-ba23-e4c9d
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('c20c69a8-7d6b-4539-9e6b-6f2c1
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('58392af6-2131-413b-b188-1851a
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('cc7cf5df-a466-42c6-9453-340aa
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('5d3b6871-20db-442e-806f-30d50
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('38a9d032-30b9-470d-93ad-47f5e
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('ba5bc2e4-b1de-4cdd-8e63-459fd
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('80db33ae-a702-41ff-8818-e72da
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('494098d5-0bdc-4e10-a30b-50818
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('7b08995d-fa1a-4ebf-98f3-a3ef3
```

Extensions that require additional software configuration:

- *Go to **Administration > Utilities > Database query**. At bottom right select option **JDBC**, **paste into the box the SQL inserts** of the extensions you want to enable. Finally click at bottom right the button **execute**.*

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('8e642680-0dcd-4ded-a501-2aa3e
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('288cdc63-78e4-434b-94e0-ccbda
```

Deprecated extensions:

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('df5eb783-fb06-4b4b-bc89-4fdaa
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('87f250d5-526d-4d8a-96ee-1e5be
```

Sample extensions for development:

There're some sample extension for development testing purpose.

- *Go to **Administration** > **Utilities** > **Database query**. At bottom right select option **JDBC**, **paste into the box the SQL inserts** of the extensions you want to enable. Finally click at bottom right the button **execute**.*

```
INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('9f84b330-d096-11df-bd3b-080020
INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('44f94470-d097-11df-bd3b-080020
INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('d95e01a0-d097-11df-bd3b-080020
INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('d9dab640-d098-11df-bd3b-080020
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Enable WebDAV

Once WebDAV is enabled, you can access the OpenKM repository with this URL:

```
http://your-domain.com:8080/OpenKM/webdav
```

*Go to **Administration > Configuration parameters**:*

Field / Property	Type	Description
system.webdav.server	**Boolean**	*Enable WebDAV service. By default it is disabled.* `true`
system.webdav.fix	**Boolean**	*Character ":" may cause problems with some WebDAV clients, Enabling this parameter will replace in the URL the character ":" by "_". By default is disabled.* ⚠ *It's recommended enabling this parameter to prevent compatibility problems with some WebDAV clients.* *A path like **/okm:root/myfile.txt** will be translated to **/okm_root/myfile.txt*** `true`

✔ *OpenKM WebDAV is based on excellent Open Source Milton WebDAV API for Java. The Milton project maintains a detailed WebDAV client compatibility list that describes a "Recipe for broad client compatibility."*

Linux

From Linux you can use:

- *http://dav.sourceforge.net/*
- *http://savannah.nongnu.org/projects/davfs2*

For example, in Ubuntu or any Debian based Linux distro you can install this utility this way:

```
$ sudo aptitude install davfs2
```

Make a mount point:

```
$ sudo mkdir ~/okmdav
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

And mount it:

```
$ sudo mount.davfs "http://your-domain.com:8080/OpenKM/webdav" ~/okmdav
```

To ease the process, add this entry to your /etc/fstab:

```
http://your-domain.com:8080/OpenKM/webdav /home/username/webdav davfs noauto,user,rw
```

*And then in the user's home directory, look for **./davfs2/davfs2.conf**. There you'll find a **if_match_bug 1** line. Uncomment it.*

See also:

- *Ubuntu: Mounting remote filesystem using davfs2 (FUSE)*

Windows

Install a WebDAV client for Windows.

We don't recommend using the default Windows WebDAV support because is buggy.

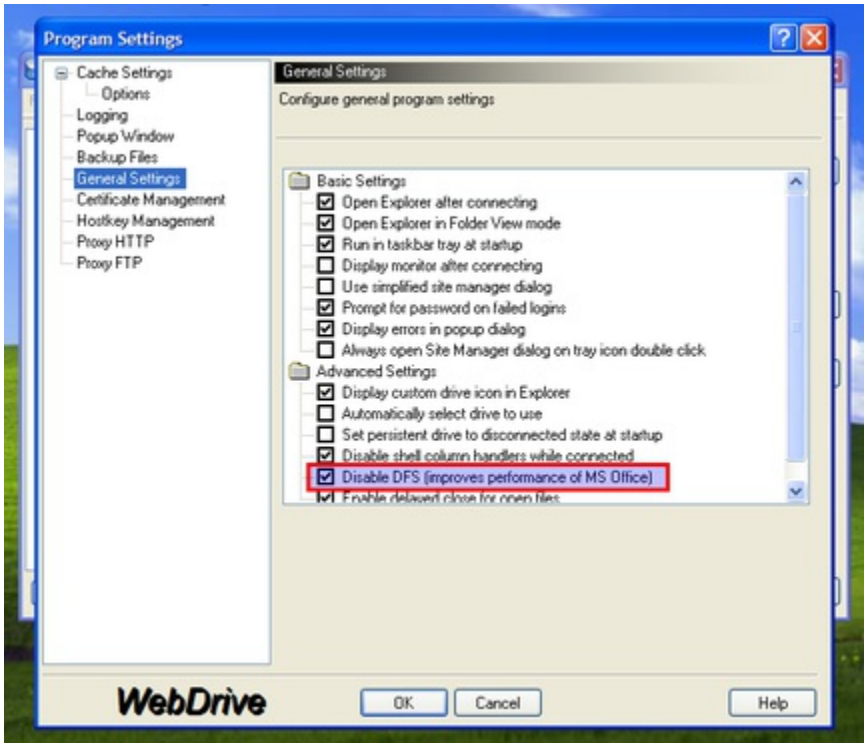
Here you have several options:

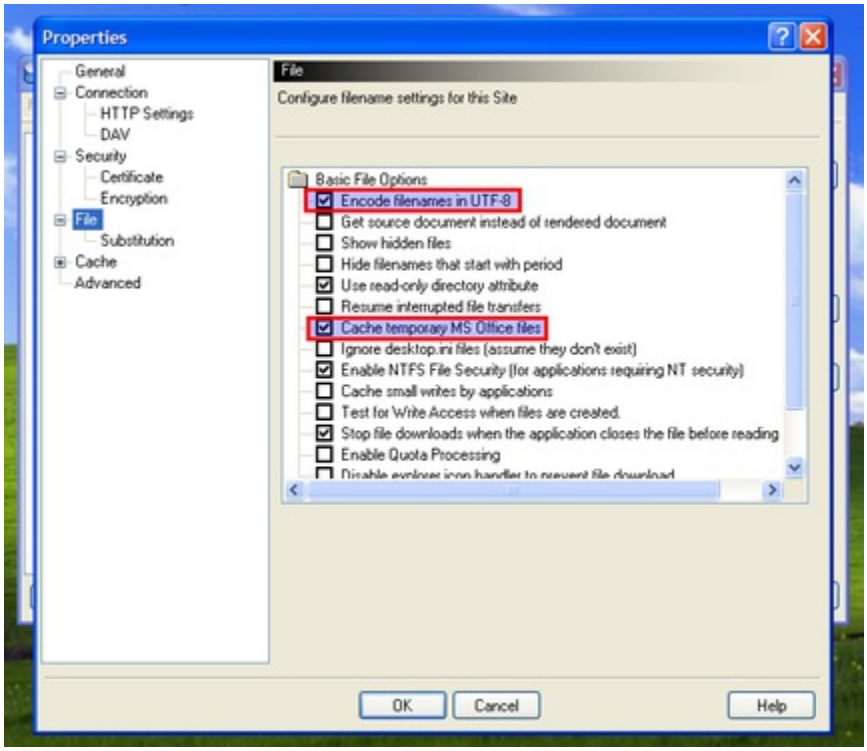
- *http://www.netdrive.net/ (Free for home use but does not handle document versioning properly)*
- *http://www.webdrive.com/ (The recommended option and will create a new document version when you save the modifications)*
- *WebDAV Clients on Windows XP*

WebDrive

Probably this is the best choice if you are under Windows, because of its stability and ease of configuration. WebDrive 10 (Version 10.00, Build 2521) has been roughly tested with OpenKM to enhance compatibility effort. This is the supported configuration to work with OpenKM WebDAV connector:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.





Windows Web Folders

Perform the following steps in Internet Explorer:

- *Select File -> Open.*

- *Fill in URL like http://your-domain.com:8080/OpenKM/webdav*
- *Check "Open as Web Folder".*
- *Click OK.*

There are some well-known issues. Please consult the following articles in case of any problem

- *Be aware that there are multiple versions of WebDAV Microsoft libraries (and different type of bugs). To avoid some frustration if these steps don't work for you Update Windows XP for Web Folders, more information at Web Folder Client (MSDAIPP.DLL) Versions and Issues.*
- *Windows Vista and Windows 7 both restrict access to WebDAV servers that use Basic HTTP authentication on non-SSL connections. This restriction can be solved by changing a registry key. Read more on. However, SSL connections do bring improved security. This page from Greenbytes is an excellent resource to analyse & fix problems with your Windows WebDAV client.*

Windows XP's "My Network Places"

Windows XP supports 'Network Places' in the standard File dialog:

- *Open networking.*
- *Connect as network resource.*
- *Write url http://your-domain.com:8080/OpenKM/webdav*
- *Set user and password.*
- *Open the Registry Editor utility (regedit.exe).*
- *Go to **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WebClient\Parameters**, look for **TypeUseBasicAuth** and replace the default value "**1**" by "**2**" allowing BasicAuth through NON SSL connections.*
- *Map your drive using this address: \\your-domain.com:8080/OpenKM/repository/default/okm_root/*

More information at our OpenKM Forum.

Other tools

- *Raiddrive*

Mac

Install any WebDAV client for Mac, here you have some options:

- *http://cyberduck.ch/*
- *http://www.webdrive.com/ (There is a version for Macintosh OS/X)*

Mac OS X finder.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Select in the Finder "**Go**" and "**Connect to Server...**".*
- *Fill in the WebDAV URL http://your-domain:8080/OpenKM/webdav*
- *Enter a username and password.*
- *And the document repository is accessible!*

Troubleshooting

Slow WedDAV performance on Windows

If you have a slow webdav connection in windows 7 please follow the instructions in Fix Slow WebDAV Performance in Windows 7. It turns out to be a windows issue, not OpenKM.

Furthermore in Windows 7/Vista you need to add your certificate (if its not issued by a verified CA) into windows if you plan to use Webdav over SSL (and,like me, do not compensate with digest auth):

1. *Get the certificate from the server administrator in CER format or download it from Firefox.*
2. *Start windows Certificate Manager: in command line type **certmgr.msc***
3. *Select on the left **Trusted Root Certification Authorities**.*
4. *In the menu **Actions** > **All Tasks** select **Import** and then **Next**.*
5. *Browse for the location of certificate file you saved earlier on your computer and click **Open**.*
6. *You will get a few more warning messages. Just say to go through with it until the certificate is accepted.*

Then map webdav content as a network drive without any other software. The folder should be: http://your-domain:8080/OpenKM/webdav

To make sure that everything is ok with certificates, open the address using IE. If no warning pops up is shown, then is right.

This allows you only to read/view files. If you need write access over a network drive you need to disable File Locking (windows does not support webdav file lock).

For more information Web Lockers Webdav For MS Windows

For Windows XP there is a fix (KB907306).

Turn off file locking

If you need to write access over a network drive you need to disable File Locking (windows does not support webdav file lock).

*Execute registry editor tool, in command line type **regedit**. Modify or create this configuration parameter:*

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WebClient\Parameters]
"SupportLocking"=dword:00000000
```

LDAP configuration

The LDAP integration is divided in two sections, login process and retrieving users and roles from LDAP server. It's usually better starting with LDAP configuration parameters which are used by OpenKM to show users and roles into administration and once this section is configured go to Spring Security configuration (openkm.xml changes).

Usually you only want to retrieve a subset of users and roles present in your LDAP, to be shown in the user interface or be able to login into OpenKM. If this is your case, please take a look at LDAP best practices for filtering users and roles .

OpenLDAP

- *OpenLDAP example with URL base at login*

Active Directory

- *Active Directory basic configuration*
- *Active Directory advanced configuration*
- *Active Directory example with referral enabled*
- *Active Directory example with login based on filtering users by roles (suggested).*
- *Active Directory example with remember-me services*
- *Active Directory mixed Configuration*

Uncategorised examples shared by users

- *https://www.nosam.com/node/8, http://pastebin.com/iQa4LhyY (more information OpenKM public forum)*

Tools

- *Apache Directory Studio.*
- *Active Directory Explorer Utility.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OpenLDAP example with URL base at login

LDAP Structure

```
dc=com
    dc=some
        ou=organization
            cn=ROLE_ADMIN
                memberUid=okmAdmin
                memberUid=user1
                memberUid=user2
            cn=ROLE_USER
                memberUid=user3
                memberUid=user4
                ...
        ou=organization
            uid=user1
                mail=user@mail.com
                cn=User Name 1
            uid=user2
                mail=user2@mail.com
                cn=User Name 3
            uid=user3
                mail=user3@mail.com
                cn=User Name 3
            uid=user4
                mail=user4@mail.com
                cn=User Name 4
```

Valid roles:

- *cn=ROLE_X,**ou=roles,dc=some,dc=com***
- *cn=ROLE_Y,ou=dept marketing,**ou=roles,dc=some,dc=com***
- *cn=ROLE_Z,ou=dept sales,**ou=roles,dc=some,dc=com***

Invalid roles:

- *cn=ROLE_INVALID,ou=dept,dc=some,dc=com (any distinguished name not included in ou=roles,dc=some,dc=com)*

Valid users:

- *uid=USER_X,**ou=organization,dc=some,dc=com***
- *uid=USER_Y,ou=dept id,**ou=organization,dc=some,dc=com***
- *uid=USER_Z,ou=dept administrator,**ou=organization,dc=some,dc=com***

Invalid users:

- *uid=USER_INVALID,ou=house,dc=some,dc=com (any distinguished name not included in ou=organization,dc=some,dc=com)*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats. In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.principal.LdapPrincipalAdapter`
system.login.lowercase	***String***	`false`
principal.ldap.server	***String***	`ldap://192.168.xxx.xxx:389`
principal.ldap.security.principal	***String***	`cn=Manager,dc=some,dc=com`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`uid`
principal.ldap.user.search.base	***List***	`ou=users,dc=some,dc=com`
principal.ldap.user.search.filter	***String***	`(objectClass=inetOrgPerson)`
principal.ldap.username.attribute	***String***	`cn`
		`ou=users,dc=some,dc=com`

principal.ldap.username.search.base	***String***	
principal.ldap.username.search.filter	***String***	`(uid={0})`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`dc=some,dc=com`
principal.ldap.mail.search.filter	***String***	`(uid={0})`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`ou=roles,dc=some,dc=com`
principal.ldap.role.search.filter	***String***	`(objectClass=posixGroup)`
principal.ldap.roles.by.user.attribute	***String***	`cn`
principal.ldap.roles.by.user.search.base	***String***	`ou=roles,dc=some,dc=com`
principal.ldap.roles.by.user.search.filter	***String***	`(memberUid={0})`
principal.ldap.users.by.role.attribute	***String***	`memberUid`
principal.ldap.users.by.role.search.base	***String***	`ou=roles,dc=some,dc=com`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=posixGroup)(cn={0}))`

Step 2 - configure login

- ***Important*** *login* ***(ldap://192.168.0.13:389/dc=some,dc=com)*** *sets default filter base queries at* ***dc=some,dc=com with is concatenated by default in all filter queries*** .
- *Roles (groups) filter base is* ***ou=roles*** *(real distinguised name is ou=roles,dc=some,dc=com). Any valid roles should have it as parent. ou=roles"/> really points to ou=roles,dc=some,dc=com"/>*
- *Users filter base is* ***ou=organization*** *(real distinguised name is ou=organization,dc=some,dc=com). Any valid user*

should have it as parent. to

- *User filter is uid={0}*
- *Finally take in consideration the value 1 at memberUid={1}"/>*

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=uuid={0}
ldap.role.search.filter=memberUid={1}
```

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [*Configuring openkm.xml*]

documentation section for more information.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta

  <security:authentication-manager alias="authenticationManager">
   <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

  <beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpri
   <beans:constructor-arg value="ldap://192.168.0.13:389/dc=some,dc=com"/>
  <beans:property name="userDn" value="cn=Manager,dc=some,dc=com"/>
   <beans:property name="password" value="******"/>
  </beans:bean>

 <beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentic
  <beans:constructor-arg>
   <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentica
```

```
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"></beans:property>
     </beans:bean>
    </beans:constructor-arg>
    <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
    </beans:bean>

     <beans:bean id="userSearch" class="org.springframework.security.ldap.search.Filter
      <beans:constructor-arg index="0" value="ou=organization" />
      <beans:constructor-arg index="1" value="uid={0}" />
      <beans:constructor-arg index="2" ref="contextSource" />
      <beans:property name="searchSubtree" value="true" />
    </beans:bean>

    <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.securit
    <beans:constructor-arg ref="contextSource"/>
    <beans:constructor-arg value="ou=roles"/>
    <beans:property name="groupSearchFilter" value="memberUid={1}"/>
    <beans:property name="groupRoleAttribute" value="cn"/>
    <beans:property name="searchSubtree" value="true" />
    <beans:property name="convertToUpperCase" value="true" />
    <beans:property name="rolePrefix" value="" />
   </beans:bean>

    <!--Needed for remember-me services -->
   <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
    <beans:constructor-arg ref="userSearch"/>
    <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
   </beans:bean>

  </beans:beans>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Active Directory example with referral enabled

This example covers the scenario where more than one Active Directory domains work together and it's needed the ***follow*** *parameter.*

LDAP Structure

```
dc=com
    dc=company
        ou=OPENKM
            cn=ROLE_ADMIN
                member=okmAdmin
                member=user1
                member=user2
            cn=ROLE_USER
                member=user3
                member=user4
            cn=ROLE_XXXX
            cn=ROLE_YYYY
                ...
        ou=organization1
            sAMAccountName=okmAdmin
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                mail=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                mail=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                mail=user2@mail.com
                cn=User Name 3
        ou=organization2
            sAMAccountName=user3
                memberOf=CN=ROLE_USER,OU=OPENKM,DC=company,DC=com
                mail=user3@mail.com
                cn=User Name 3
            sAMAccountName=user4
                memberOf=CN=ROLE_USER,OU=OPENKM,DC=company,DC=com
                mail=user4@mail.com
                cn=User Name 4
```

Valid groups:

- *cn=ROLE_ADMIN,ou=OPENKM,dc=company,dc=com*
- *cn=ROLE_USER,ou=OPENKM,dc=company,dc=com*
- *cn=ROLE_XXXX,ou=OPENKM,dc=company,dc=com*
- *cn=ROLE_YYYY,ou=OPENKM,dc=company,dc=com*

Valid users:

- *cn=user1,ou=organization1,dc=company,dc=com*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *cn=user2,ou=organization1,dc=company,dc=com*
- *cn=user3,ou=organization2,dc=company,dc=com*
- *cn=user4,ou=organization2,dc=company,dc=com*

Any distinguished name includes by default ***dc=company,dc=com***

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats. In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l****ogin in OpenKM with the admin URL*** *(for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

The first action should be to modify ***principal.adapter*** *parameter value and* ***restart OpenKM service****. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

Go to ***Administration > Configuration parameters****:*

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdapt`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.xxx.xxx:389`
principal.ldap.security.principal	***String***	`CN=Administrator,OU=OPENKM,DC=company,DC=com`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	`follow`
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`DC=company,DC=com`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.user.search.filter	***String***	`(objectclass=user)`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`DC=company,DC=com`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`DC=company,DC=com`
principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`DC=company,DC=com`
principal.ldap.role.search.filter	***String***	`(objectclass=group)`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`DC=company,DC=com`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(CN={0}))`

Step 2 - configure login

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Parameter **follow** indicates several domains servers working together (balanced).*
- *Users defined in any Active Directory node will be able to login, because it has defined DC=company,DC=com as base filter.*
- *Any user authenticated in Active Directory can login because has not any filtering clausule.*
- *Groups read by OpenKM can be defined in any Active Directory node, because it has defined DC=company,DC=com as base filter.*

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=sAMAccountName={0}
ldap.role.search.filter=member={0}
```

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [*Configuring openkm.xml*]

documentation section for more information.

```
<security:authentication-manager alias="authenticationManager">
    <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

  <beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpri
    <beans:constructor-arg value="ldap://192.168.xxx.xxx:389"/>
    <beans:property name="userDn" value="CN=Administrator,OU=OPENKM,DC=company,DC=com
    <beans:property name="password" value="****"/>
    <beans:property name="baseEnvironmentProperties">
      <beans:map>
        <beans:entry>
          <beans:key>
            <beans:value>java.naming.referral</beans:value>
          </beans:key>
          <beans:value>follow</beans:value>
        </beans:entry>
      </beans:map>
    </beans:property>
  </beans:bean>
```

```
<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authenti
  <beans:constructor-arg>
    <beans:bean class="org.springframework.security.ldap.authentication.BindAuthent
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"/>
    </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPop
</beans:bean>

<beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterB
  <beans:constructor-arg index="0" value="DC=company,DC=com" />
  <beans:constructor-arg index="1" value="sAMAccountName={0}" />
  <beans:constructor-arg index="2" ref="contextSource" />
  <beans:property name="searchSubtree" value="true" />
</beans:bean>

<beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.securit
<beans:constructor-arg ref="contextSource"/>
<beans:constructor-arg value="DC=openkm,DC=local"/>
<beans:property name="groupSearchFilter" value="member={0}"/>
<beans:property name="groupRoleAttribute" value="cn"/>
<beans:property name="searchSubtree" value="true" />
<beans:property name="convertToUpperCase" value="false" />
<beans:property name="rolePrefix" value="" />
</beans:bean>

<!--Needed for remember-me services -->
<beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
<beans:constructor-arg ref="userSearch"/>
<beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
</beans:bean>
```

Active Directory example with login based on filtering users by roles

In this example, we'll see an Active Directory connection that allows to connect any Active Directory authenticated user member of ROLE_USER or ROLE_ADMIN.

LDAP structure

```
dc=com
    dc=company
        ou=OPENKM
            cn=ROLE_ADMIN
                member=okmAdmin
                member=user1
            cn=ROLE_USER
                member=user3
            cn=ROLE_XXXX
                member=user2
            cn=ROLE_YYYY
                member=user4
                ...
        ou=organization1
            sAMAccountName=okmAdmin
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                userPrincipalName=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                memberOf=CN=ROLE_XXXX,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user2@mail.com
                cn=User Name 3
        ou=organization2
            sAMAccountName=user3
                memberOf=CN=ROLE_USER,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user3@mail.com
                cn=User Name 3
            sAMAccountName=user4
                memberOf=CN=ROLE_YYYY,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user4@mail.com
                cn=User Name 4
```

Valid groups:

- *cn=ROLE_ADMIN,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_USER,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_XXXX,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_YYYY,**ou=OPENKM,dc=company,dc=com***

Valid users:

- *cn=user1,**ou=organization1,dc=company,dc=com***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *cn=user2,**ou=organization1,dc=company,dc=com***

Invalid users:

- *cn=user3,**ou=organization2,dc=company,dc=com***
- *cn=user4,**ou=organization2,dc=company,dc=com***

*Any distinguished name includes by default dc=company,dc=com except users which are not member of **ROLE_ADMIN** or **ROLE_USER***

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats . In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

- *User members of ROLE_ADMIN or ROLE_USER can be created in any active directory node, because it has defined DC=company,DC=com as base filter, **principal.ldap.user.search.base=DC=company,DC=com** and user search filter as **principal.ldap.user.search.filter=(&(objectclass=user)(|(memberOf=CN=ROLE_ADMIN,OU=OpenKM,DC=company,DC=com)***.
- *Groups can be created in any active directory node, because is has DC=company,DC=com as base filter, **principal.ldap.role.search.base=DC=company,DC=com***.
- *All Active Directory groups will be listed, because it has not applied any filter restriction **principal.ldap.role.search.filter=(objectclass=group).***
- *Mail attribute used in this case is **userPrincipalName** (the most usual ldap attribute to get mail is **mail**) **principal.ldap.mail.attribute=userPrincipalName***.

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdap`

system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.xxx.xxx:389`
principal.ldap.security.principal	***String***	`CN=Administrator,OU=OPENKM,DC=company,DC=com`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	`follow`
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`DC=company,DC=com`

principal.ldap.user.search.filter	***String***	`(&(objectclass=user)(|`
		`(memberOf=CN=ROLE_ADMIN,OU=OpenKM,DC=company,`
		`(memberOf=CN=ROLE_USER,OU=OpenKM,DC=company,D`

principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`DC=company,DC=com`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`userPrincipalName`
principal.ldap.mail.search.base	***String***	`DC=company,DC=com`
principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`DC=company,DC=com`
principal.ldap.role.search.filter	***String***	`(objectclass=group)`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`OU=OpenKM,DC=company,DC=com`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(cn={0}))`

Step 2 - configure login

- *Users created in any Active Directory node members of ROLE_ADMIN or ROLE_USER will be able to login, because it has defined DC=company,DC=com as base filter, and* ***as user account filter.***
- *Groups read by OpenKM can be created in any Active Directory node, because it has defined DC=company,DC=com as base filter.*

Apply changes in the ***openkm.properties*** *file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=(&(sAMAccountName={0})(|(memberOf=CN=ROLE_ADMIN,OU=OpenKM,DC=
ldap.role.search.filter=member={0}
```

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

In some cases might be interested to set configuration in file openkm.xml please refer to Configuring openkm.xml documentation section for more information.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta

<security:authentication-manager alias="authenticationManager">
     <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

<beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpring
  <beans:constructor-arg value="ldap://192.168.xxx.xxx"/>
  <beans:property name="userDn" value="CN=connect,OU=OpenKM,DC=company,DC=com"/>
  <beans:property name="password" value="*****"/>
  <beans:property name="baseEnvironmentProperties">
     <beans:map>
        <beans:entry>
          <beans:key>
            <beans:value>java.naming.referral</beans:value>
          </beans:key>
          <beans:value>follow</beans:value>
        </beans:entry>
      </beans:map>
   </beans:property>
</beans:bean>

<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentica
  <beans:constructor-arg>
    <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentic
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"/>
    </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
</beans:bean>

<beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterBas
  <beans:constructor-arg index="0" value="DC=company,DC=com" />
  <beans:constructor-arg index="1" value="(&amp;(sAMAccountName={0})(|(memberOf=CN=RO
  <beans:constructor-arg index="2" ref="contextSource" />
  <beans:property name="searchSubtree" value="true" />
</beans:bean>

 <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.security
  <beans:constructor-arg ref="contextSource"/>
      <beans:constructor-arg value="DC=company,DC=com"/>
      <beans:property name="groupSearchFilter" value="member={0}"/>
      <beans:property name="groupRoleAttribute" value="cn"/>
      <beans:property name="searchSubtree" value="true" />
      <beans:property name="convertToUpperCase" value="false" />
      <beans:property name="rolePrefix" value="" />
 </beans:bean>

  <!--Needed for remember-me services -->
 <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
```

```
    <beans:constructor-arg ref="userSearch"/>
    <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
  </beans:bean>

</beans:beans>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Active Directory basic configuration

This is the suggested configuration to be used when roles and users are both defined in the same node, otherwise refer to other example.

LDAP structure

```
dc=com
    dc=company
        cn=users
            cn=ROLE_ADMIN
                member=okmAdmin
                member=user1
            cn=ROLE_USER
                member=user3
            cn=ROLE_XXXX
                member=user2
            cn=ROLE_YYYY
                member=user4
                ...
            sAMAccountName=okmAdmin
                memberOf=cn=ROLE_ADMIN,cn=users,dc=company,dc=com
                userPrincipalName=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                memberOf=cn=ROLE_ADMIN,cn=users,dc=company,dc=com
                userPrincipalName=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                memberOf=cn=ROLE_USER,cn=users,dc=company,dc=com
                userPrincipalName=user2@mail.com
                cn=User Name 3
```

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats . In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

- *User members of ROLE_ADMIN or ROLE_USER are created into users node (these should be the distinguished names **CN=ROLE_ADMIN,cn=users,DC=company,DC=com** and **CN=ROLE_USER,cn=users,DC=company,DC=com)**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Groups can be created on any Active Directory node, because DC=company,DC=com is set as base filter,* ***principal.ldap.role.search.base=DC=company,DC=com****.*
- *All Active Directory groups will be listed, because it has not applied any filter restriction* ***principal.ldap.role.search.filter=(objectclass=group).***

Go to Administration > Configuration parameters:

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdapt`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.0.6:389`
principal.ldap.security.principal	***String***	`CN=Administrator,cn=users,dc=company,dc=local`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`cn=users,dc=company,dc=local`
principal.ldap.user.search.filter	***String***	`(objectclass=person)`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`cn=users,dc=company,dc=local`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`cn=users,dc=company,dc=local`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`cn=users,dc=company,dc=local`
principal.ldap.role.search.filter	***String***	`(objectclass=group)`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`cn={0},cn=users,dc=company,dc=local`
principal.ldap.users.by.role.search.filter	***String***	`(objectclass=group)`

Step 2 - configure login

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=sAMAccountName={0}
ldap.role.search.filter=member={0}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [Configuring openkm.xml](#)

documentation section for more information.

```
<security:ldap-server id="ldapServer"
  url="ldap://192.168.0.6:389/DC=ldap,dc=company,dc=local"
  manager-dn="CN=Administrator,cn=users,dc=company,dc=local"
  manager-password="password"/>

<security:authentication-manager alias="authenticationManager">
  <security:ldap-authentication-provider
    server-ref="ldapServer"
    user-search-base="cn=Users"
    user-search-filter="(sAMAccountName={0})"
    group-search-base="cn=Users"
    group-search-filter="(member={0})"
    group-role-attribute="cn"
    role-prefix="none">
  </security:ldap-authentication-provider>
</security:authentication-manager>

<beans:bean id="userDetailService" class="com.openkm.spring.UserTenantDetailsService"
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Active Directory advanced configuration

LDAP structure

```
dc=com
    dc=company
        cn=users
            cn=ROLE_ADMIN
                member=okmAdmin
                member=user1
            cn=ROLE_USER
                member=user3
            cn=ROLE_XXXX
                member=user2
            cn=ROLE_YYYY
                member=user4
                ...
            sAMAccountName=okmAdmin
                memberOf=cn=ROLE_ADMIN,cn=users,dc=company,dc=com
                userPrincipalName=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                memberOf=cn=ROLE_ADMIN,cn=users,dc=company,dc=com
                userPrincipalName=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                memberOf=cn=ROLE_USER,cn=users,dc=company,dc=com
                userPrincipalName=user2@mail.com
                cn=User Name 3
```

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats . In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

- *User members of ROLE_ADMIN or ROLE_USER are created into users node (these should be the distinguished names **CN=ROLE_ADMIN,cn=users,DC=company,DC=com** and **CN=ROLE_USER,cn=users,DC=company,DC=com)**.*
- *Groups can be created on any Active Directory node, because DC=company,DC=com is set as base filter, **principal.ldap.role.search.base=DC=company,DC=com**.*
- *All Active Directory groups will be listed, because has not applied any filter restriction*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.role.search.filter=(objectclass=group).

Go to ***Administration > Configuration parameters****:*

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdapt`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.0.6:389`
principal.ldap.security.principal	***String***	`CN=Administrator,cn=users,dc=company,dc=local`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`dc=company,dc=local`
principal.ldap.user.search.filter	***String***	`(objectclass=person)`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`dc=company,dc=local`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`dc=company,dc=local`
principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`dc=company,dc=local`
principal.ldap.role.search.filter	***String***	`(objectclass=group)`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`dc=company,dc=local`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(cn={0}))`

Step 2 - configure login

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=sAMAccountName={0}
ldap.role.search.filter=member={0}
```

After the update of the openkm.properties file must restart the openkm service to take effect.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [*Configuring openkm.xml*]

documentation section for more information.

```
<security:authentication-manager alias="authenticationManager">
  <security:authentication-provider ref="ldapAuthProvider" />
</security:authentication-manager>

<beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpring
  <beans:constructor-arg value="ldap://192.168.0.6:389/dc=company,dc=local"/>
  <beans:property name="userDn" value="CN=Administrator,cn=users,dc=company,dc=local"
  <beans:property name="password" value="password"/>
</beans:bean>

<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentica
  <beans:constructor-arg>
    <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentic
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"/>
    </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
</beans:bean>

<beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterBas
  <beans:constructor-arg index="0" value="" />
  <beans:constructor-arg index="1" value="sAMAccountName={0}" />
  <beans:constructor-arg index="2" ref="contextSource" />
  <beans:property name="searchSubtree" value="true" />
</beans:bean>

 <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.security
  <beans:constructor-arg ref="contextSource"/>
      <beans:constructor-arg value=""/>
      <beans:property name="groupSearchFilter" value="memberOf={1}"/>
      <beans:property name="groupRoleAttribute" value="cn"/>
      <beans:property name="searchSubtree" value="true" />
      <beans:property name="convertToUpperCase" value="false" />
      <beans:property name="rolePrefix" value="" />
 </beans:bean>

  <!--Needed for remember-me services -->
 <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
  <beans:constructor-arg ref="userSearch"/>
  <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
 </beans:bean>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Active Directory mixed Configuration

The objective of this kind of configuration is to be able to authenticate against Active Directory, but on the other hand, obtain roles from the database.

Available since OpenKM 6.4.28 version

LDAP structure

```
dc=com
    dc=company
        cn=users
            sAMAccountName=okmAdmin
                cn=users,dc=company,dc=com
                userPrincipalName=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                cn=users,dc=company,dc=com
                userPrincipalName=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                cn=users,dc=company,dc=com
                userPrincipalName=user2@mail.com
                cn=User Name 3
```

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats . In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

*We **suggest** to use "**com.openkm.plugin.principal.DatabasePrincipalAdapter**" in the **principal.adapter** configuration parameter value, either "**com.openkm.plugin.principal.MixedPrincipalAdapter**" value shown in the table below.*

*If you use "**DatabasePrincipalAdapter**" the restrictions are:*

- *Before user login, you must create in the database all the user data.*

- *The password you set in the database will not be applied, the login process works only with AD connection.*

- *User members of ROLE_ADMIN or ROLE_USER are created into the database.*

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.MixedPrincipalAdapte`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.0.6:389`
principal.ldap.security.principal	***String***	`CN=Administrator,cn=users,dc=company,dc=local`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`dc=company,dc=local`
principal.ldap.user.search.filter	***String***	`(objectclass=person)`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`dc=company,dc=local`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`dc=company,dc=local`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`

Step 1 - configure login

You must disable default OpenKM authentication to enable openkm.xml

You must set the location of the openkm.xml file

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to disable internal ldap login.*

```
# Disable default OpenKM authentication
okm.authentication.database=false
okm.authentication.ldap=false

# Enable openkm.xml configuration file
okm.authentication.config=file:/home/openkm/tomcat-8.5.34/openkm.xml
```

*Apply changes in the **openkm.xml** file (refer to Configuring openkm.xml documentation section for more information).*

```
  <!-- LDAP authentication -->
  <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

  <beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpri
    <beans:constructor-arg value="ldap://192.168.0.6:389"/>
    <beans:property name="userDn" value="CN=Administrator,cn=users,dc=company,dc=loca
    <beans:property name="password" value="password"/>
    <beans:property name="baseEnvironmentProperties">
       <beans:map>
          <beans:entry>
             <beans:key>
              <beans:value>java.naming.referral</beans:value>
            </beans:key>
            <beans:value>follow</beans:value>
          </beans:entry>
       </beans:map>
    </beans:property>
  </beans:bean>

  <beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authenti
    <beans:constructor-arg>
      <beans:bean class="org.springframework.security.ldap.authentication.BindAuthent
        <beans:constructor-arg ref="contextSource"/>
        <beans:property name="userSearch" ref="userSearch"/>
      </beans:bean>
    </beans:constructor-arg>
    <beans:constructor-arg name="authoritiesPopulator" ref="databaseAuthoritiesPopula
  </beans:bean>

  <beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterB
    <beans:constructor-arg index="0" value="dc=company,dc=local" />
    <beans:constructor-arg index="1" value="(uid={0})" />
```

```
    <beans:constructor-arg index="2" ref="contextSource" />
    <beans:property name="searchSubtree" value="true" />
  </beans:bean>

  <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdet
    <beans:constructor-arg ref="userSearch"/>
    <beans:constructor-arg ref="databaseAuthoritiesPopulator"/>
  </beans:bean>

  <!-- Get roles from database -->
  <beans:bean id="databaseAuthoritiesPopulator" class="com.openkm.DatabaseAuthorities
    <beans:constructor-arg ref="dataSource"/>
    <beans:constructor-arg value="select ur_user, ur_role from OKM_USER_ROLE where ur
  </beans:bean>
```

After the changes in the previous files must restart the openkm service to take effect.

Active Directory example with remember-me services

In this example, we'll see an Active Directory connection that allows to connect any Active Directory authenticated user member of ROLE_USER or ROLE_ADMIN as the same time as using remeber me services.

Available since OpenKM 6.4.25 version

LDAP structure

```
dc=com
    dc=company
        ou=OPENKM
            cn=ROLE_ADMIN
                member=okmAdmin
                member=user1
            cn=ROLE_USER
                member=user3
            cn=ROLE_XXXX
                member=user2
            cn=ROLE_YYYY
                member=user4
                ...
        ou=organization1
            sAMAccountName=okmAdmin
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                userPrincipalName=okmAdmin@mail.com
                cn=OpenKM Administrator
            sAMAccountName=user1
                memberOf=CN=ROLE_ADMIN,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user1@mail.com
                cn=User Name 1
            sAMAccountName=user2
                memberOf=CN=ROLE_XXXX,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user2@mail.com
                cn=User Name 3
        ou=organization2
            sAMAccountName=user3
                memberOf=CN=ROLE_USER,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user3@mail.com
                cn=User Name 3
            sAMAccountName=user4
                memberOf=CN=ROLE_YYYY,OU=OPENKM,DC=company,DC=com
                userPrincipalName=user4@mail.com
                cn=User Name 4
```

Valid groups:

- *cn=ROLE_ADMIN,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_USER,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_XXXX,**ou=OPENKM,dc=company,dc=com***
- *cn=ROLE_YYYY,**ou=OPENKM,dc=company,dc=com***

Valid users:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *cn=user1,**ou=organization1,dc=company,dc=com***
- *cn=user2,**ou=organization1,dc=company,dc=com***

Invalid users:

- *cn=user3,**ou=organization2,dc=company,dc=com***
- *cn=user4,**ou=organization2,dc=company,dc=com***

*Any distinguished name includes by default dc=company,dc=com except users which are not member of **ROLE_ADMIN** or **ROLE_USER***

The OpenKM integration with LDAP has two steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats . In the second step configure login, this configuration works always in real time.

Step 1 - configuration parameters

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

- *User members of ROLE_ADMIN or ROLE_USER can be created in any active directory node, because it has defined DC=company,DC=com as base filter, **principal.ldap.user.search.base=DC=company,DC=com** and user search filter as **principal.ldap.user.search.filter=(&(objectclass=user)(|(memberOf=CN=ROLE_ADMIN,OU=OpenKM,DC=company,DC=com)***.
- *Groups can be created in any active directory node, because is has DC=company,DC=com as base filter, **principal.ldap.role.search.base=DC=company,DC=com***.
- *All Active Directory groups will be listed, because it has not applied any filter restriction **principal.ldap.role.search.filter=(objectclass=group).***
- *Mail attribute used in this case is **userPrincipalName** (the most usual ldap attribute to get mail is **mail**) **principal.ldap.mail.attribute=userPrincipalName***.

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*

principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdap`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.xxx.xxx:389`
principal.ldap.security.principal	***String***	`CN=Administrator,OU=OPENKM,DC=company,DC=com`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	`follow`
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`DC=company,DC=com`

principal.ldap.user.search.filter	***String***	`(&(objectclass=user)(|`
		`(memberOf=CN=ROLE_ADMIN,OU=OpenKM,DC=company,`
		`(memberOf=CN=ROLE_USER,OU=OpenKM,DC=company,D`

principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`DC=company,DC=com`
principal.ldap.username.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.mail.attribute	***String***	`userPrincipalName`
principal.ldap.mail.search.base	***String***	`DC=company,DC=com`
principal.ldap.mail.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`DC=company,DC=com`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.role.search.filter	***String***	`(objectclass=group)`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`OU=OpenKM,DC=company,DC=com`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(cn={0}))`

Step 2 - configure login

- *Users created in any Active Directory node members of ROLE_ADMIN or ROLE_USER will be able to login, because it has defined DC=company,DC=com as base filter, and* ***as user account filter.***
- *Groups read by OpenKM can be created in any Active Directory node, because it has defined DC=company,DC=com as base filter.*
- *DefaultLdapAuthorities populator has been extracted to an independent bean in order to be able to use it in userDetailService Bean as well as ldapAuthProvider bean.*

Apply changes in the ***openkm.properties*** *file.*

*The parameter "****okm.authentication.database****" is used to disable database login.*

*The parameter "****okm.authentication.ldap****" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
ldap.server=ldap://192.168.0.13
ldap.manager.distinguished.name=CN=Administrator,CN=Users,DC=openkm,DC=local
ldap.manager.password=*secret*
ldap.base=DC=openkm,DC=local
ldap.role.attribute=cn
ldap.user.search.filter=sAMAccountName={0}
ldap.role.search.filter=member={0}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [Configuring openkm.xml](#)

documentation section for more information.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta

<security:authentication-manager alias="authenticationManager">
     <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

<beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpring
  <beans:constructor-arg value="ldap://192.168.xxx.xxx"/>
  <beans:property name="userDn" value="CN=connect,OU=OpenKM,DC=company,DC=com"/>
  <beans:property name="password" value="*****"/>
  <beans:property name="baseEnvironmentProperties">
     <beans:map>
        <beans:entry>
          <beans:key>
            <beans:value>java.naming.referral</beans:value>
          </beans:key>
          <beans:value>follow</beans:value>
        </beans:entry>
      </beans:map>
   </beans:property>
</beans:bean>

<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentica
  <beans:constructor-arg>
    <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentic
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"/>
    </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
</beans:bean>

<beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.security.
 <beans:constructor-arg ref="contextSource"/>
 <beans:constructor-arg value="DC=company,DC=com"/>
 <beans:property name="groupSearchFilter" value="member={0}"/>
 <beans:property name="groupRoleAttribute" value="cn"/>
 <beans:property name="searchSubtree" value="true" />
 <beans:property name="convertToUpperCase" value="false" />
 <beans:property name="rolePrefix" value="" />
</beans:bean>

 <beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterBa
   <beans:constructor-arg index="0" value="DC=company,DC=com" />
   <beans:constructor-arg index="1" value="(&amp;(sAMAccountName={0})(|(memberOf=CN=R
```

```
    <beans:constructor-arg index="2" ref="contextSource" />
    <beans:property name="searchSubtree" value="true" />
  </beans:bean>

  <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
   <beans:constructor-arg ref="userSearch"/>
   <beans:constructor-arg ref="databaseAuthoritiesPopulator"/>
  </beans:bean>

 </beans:beans>
```

LDAP best practices for filtering users and roles

Usually you only want to retrieve a subset of users and roles present in your LDAP, to be shown in user interface lists or be able to login into.

Example based on single LDAP group

- *Create a group named **OpenKM (CN=OpenKM,CN=users,DC=company,DC=com)**. It will be used for filtering **users and roles** purpose.*
- *Assign roles and users are members of OpenKM group.*
- *Goal, only users and roles which are members of OpenKM group will be displayed in user interface lists.*

For filtering user interface:

Field / Property	*Type*	*Description*
principal.ldap.user.search.filter	***String***	`(&(objectclass=person)(memberOf=CN=OpenKM,CN=users,dc=company,dc=com)`
principal.ldap.role.search.filter	***String***	`(&(objectclass=group)(memberOf=CN=OpenKM,CN=users,dc=company,dc=com)`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectclass=group)(cn={0})(memberOf=CN=OpenKM,CN=users,dc=company,dc=com)`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0})(memberOf=CN=OpenKM,CN=users,dc=company,dc=com)`

For login restriction:

Modify openkm.properties with:

```
ldap.user.search.filter=(&(sAMAccountName={0})(memberOf=CN=OpenKM,CN=users,DC=company
```

In case you use openkm.xml then with:

```
<module-option name="baseFilter">(&amp;(sAMAccountName={0})(memberOf=CN=OpenKM,CN=use
```

Example based on ROLE_USER and ROLE_ADMIN group

- *Create group name **OpenKM (CN=OpenKM,CN=users,DC=company,DC=com)**. It will be used for filtering **roles***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

purpose.

- *Assign roles are members of OpenKM group.*
- *Goal, only roles with OpenKM role will be displayed in user interface lists.*
- *Goal, only users what are members of ROLE_USER or ROLE_ADMIN group will be displayed in user interface lists.*

For filtering user interface:

Field / Property	*Type*	*Description*

principal.ldap.user.search.filter	***String***	`(&(objectclass=person) (|`
		`(memberOf=cn=ROLE_USER,cn=users,dc=company,dc=`
		`(memberOf=cn=ROLE_ADMIN,cn=users,dc=company,dc=`

principal.ldap.role.search.filter	***String***	`(&(objectclass=group)` `(memberOf=cn=OpenKM,dc=company,dc=com))`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(cn={0})` `(memberOf=cn=OpenKM,cn=users,DC=company,DC=com`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(sAMAccountName={0})` `(memberOf=cn=OpenKM,cn=users,DC=company,DC=com`

For login restriction:

Modify openkm.properties with:

```
ldap.user.search.filter=(&(sAMAccountName={0})(objectClass=person)(|(memberOf=CN=ROLE_
```

In case you use openkm.xml then with:

```
<module-option name="baseFilter">(&amp;(sAMAccountName={0})(objectClass=person)(|(mem
```

There are some characters restricted in XML, one of these is ***&****. Should be used* ***&*** *in place you usually should use* ***&****, otherwise you will get an error when starting the application.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ClearOS LDAP configuration

Need to install the Certificate Manager and Directory Server modules.

- *Create an **okmAdmin** user in ClearOS.*
- *Create **ROLE_ADMIN, ROLE_USER** groups in ClearOS, populate.*

LDAP Structure

```
dc=local
    dc=mydomain
     ou=Accounts
      ou=Groups
       cn=ROLE_ADMIN
        member=okmAdmin
        member=user1
        member=user2
        objectClass=posixGroup
       cn=ROLE_USER
        member=user3
        member=user4
        objectClass=posixGroup
       cn=ROLE_XXXX
        objectClass=posixGroup
       cn=ROLE_YYYY
        objectClass=posixGroup
       ...

                  ou=Users
       uid=okmAdmin
        memberOf=CN=ROLE_ADMIN,ou=Groups,ou=Accounts,dc=mydomain,dc=local
        mail=okmAdmin@mail.com
        cn=OpenKM Administrator
        objectClass=inetOrgPerson
       uid=user1
        memberOf=CN=ROLE_ADMIN,ou=Groups,ou=Accounts,dc=mydomain,dc=local
        mail=user1@mail.com
        cn=User Name 1
        objectClass=inetOrgPerson
       uid=user2
        memberOf=CN=ROLE_ADMIN,ou=Groups,ou=Accounts,dc=mydomain,dc=local
        mail=user2@mail.com
        cn=User Name 3
        objectClass=inetOrgPerson
```

*Any distinguished name include by default **dc=mydomain,dc=local***

The OpenKM integration with LDAP has three steps. In the first step, configure OpenKM to retrieve the list of users and roles from the LDAP. This list is cached during 30-45 minutes by OpenKM to prevent overloading the LDAP server. You can clean the cache from Administration > Tools > Cache stats. In the second step create the certificate and finally in the third step configure login, this last configuration works always in real time.

Step 1 - configuration parameters

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*We suggest l**ogin in OpenKM with the admin URL** (for example http://localhost:8080/openkm/admin/index) because in the next steps will be necessary to restart OpenKM service and you do not want to lose administration access.*

*The first action should be to modify **principal.adapter** parameter value and **restart OpenKM service**. Because session ID is kept in the browser you should not lose the login after the service restarted and can continue working in the administration. After this change, the users and roles list will be empty from the administration. Until success configuring the next parameters these lists will be empty.*

Go to Administration** > **Configuration parameters

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdapt`
principal.database.filter.inactive.users	***Boolean***	`true`
principal.hide.connection.roles	***Boolean***	`false`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldaps://ldap.mydomain.local:636`
principal.ldap.security.principal	***String***	`cn=Manager,ou=Internal,dc=mydomain,dc=local`
principal.ldap.security.credentials	***String***	`pass1234`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`uid`
principal.ldap.user.search.base	***List***	`ou=Users,ou=Accounts,dc=mydomain,dc=local`
principal.ldap.user.search.filter	***String***	`(objectClass=inetOrgPerson)`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`ou=Users,ou=Accounts,dc=mydomain,dc=local`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.username.search.filter	***String***	`(&(objectclass=inetOrgPerson)(uid={0}))`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`dc=company,dc=local`
principal.ldap.mail.search.filter	***String***	`(&(objectclass=inetOrgPerson)(uid={0}))`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`ou=Groups,ou=Accounts,dc=mydomain,dc=local`
principal.ldap.role.search.filter	***String***	`objectClass=posixGroup`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`ou=Users,ou=Accounts,dc=mydomain,dc=local`
principal.ldap.roles.by.user.search.filter	***String***	`(&(objectClass=person)(uid={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`ou=Groups,ou=Accounts,dc=mydomain,dc=local`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=posixGroup)(cn={0}))`

Step 2 - create a Certificate

ClearOS by default only allows LDAPS.

> *You should create a certificate on ClearOS with the leftmost RDN of the Subject being the common name (FQDN) of your ClearOS server. Easy way is to only specify the common name, and when signing use a policy that is permissive.*
>
> *You can create an alias (CNAME) called ldap if you wish.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

$ openssl req -out ldap.csr -key private/sys-0-key.pem -new

$ openssl ca -policy policy_anything -days 3650 -out /etc/pki/CA/certs/ldap.mydomain.local.crt -infiles ldap.csr

Add the CA cert to the nssdb:

$ certutil -A -d /etc/pki/nssdb/ -n "CA certificate" -t "CT,," -a -i /etc/pki/CA/ca-cert.pem

The LDAP server certificate needs to be saved in the nssdb with the name of Server-Cert.

$ openssl pkcs12 -export -inkey /etc/pki/CA/private/sys-0-key.pem -in /etc/pki/CA/certs/ldap.mydomain.local.crt -out

/root/Server-Cert.p12 -nodes -name 'Server-Cert'

$ pk12util -i /root/Server-Cert.p12 -d /etc/pki/nssdb/

Edit the file/etc/openldap/slapd.conf:

$ vim /etc/openldap/slapd.conf

```
TLSCACertificatePath    /etc/pki/nssdb
TLSCertificateFile      Server-Cert
TLSVerifyClient         never
```

Allow LDAP in ClearOS to be queried by OpenKM - edit the end of slapd.conf thusly:

```
access to *
    by self write
    by peername.ip=127.0.0.1 read
    ...
    by peername.ip=<OpenKM's IP address> read
```

Copy ClearOS' public CA certificate to the OpenKM server and add it to OpenKM's keystore:

$ keytool -import -trustcacerts -keystore /opt/openkm-6.2.3-community/java/jre/lib/security/cacerts -alias myca -file

/etc/pki/tls/certs/myca.pem

Step 3 - configure login

*Apply changes in the **openkm.properties** file.*

*The parameter "**okm.authentication.database**" is used to disable database login.*

*The parameter "**okm.authentication.ldap**" is used to enable ldap login.*

```
#Authentication
okm.authentication.database=false
okm.authentication.ldap=true

#LDAP
```

```
ldap.server=ldaps://ldap.mydomain.local
ldap.manager.distinguished.name=cn=Manager,ou=Internal,dc=mydomain,dc=local
ldap.manager.password=*secret*
ldap.base=dc=mydomain,dc=local
ldap.role.attribute=cn
ldap.user.search.filter=uid={0}
ldap.role.search.filter=member={0}
```

After the update of the openkm.properties file must restart the openkm service to take effect.

Another option to configure login

In some cases might be interested to set configuration in file openkm.xml please refer to [Configuring openkm.xml](#)

documentation section for more information.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta


  <!-- Security configuration -->
  <security:authentication-manager alias="authenticationManager">
   <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>

  <beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpri
     <beans:constructor-arg value="ldaps://ldap.mydomain.local:636/dc=mydomain,dc=loc
    <beans:property name="userDn" value="cn=Manager,ou=Internal,dc=mydomain,dc=local"
     <beans:property name="password" value="*secret*"/>
  </beans:bean>

  <beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authenti
 <beans:constructor-arg>
  <beans:bean class="org.springframework.security.ldap.authentication.BindAuthenticat
   <beans:constructor-arg ref="contextSource"/>
   <beans:property name="userSearch" ref="userSearch"></beans:property>
    </beans:bean>
 </beans:constructor-arg>
 <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopula
  </beans:bean>

   <beans:bean id="userSearch" class="org.springframework.security.ldap.search.Filter
       <beans:constructor-arg index="0" value="ou=Users,ou=Accounts" />
       <beans:constructor-arg index="1" value="uid={0}" />
       <beans:constructor-arg index="2" ref="contextSource" />
       <beans:property name="searchSubtree" value="true" />
   </beans:bean>

   <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.securi
  <beans:constructor-arg ref="contextSource"/>
   <beans:constructor-arg value="ou=groups,ou=Accounts"/>
   <beans:property name="groupSearchFilter" value="member={0}"/>
```

```
    <beans:property name="groupRoleAttribute" value="cn"/>
    <beans:property name="searchSubtree" value="true" />
    <beans:property name="convertToUpperCase" value="true" />
    <beans:property name="rolePrefix" value="" />
  </beans:bean>

  <!--Needed for remember-me services -->
  <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
   <beans:constructor-arg ref="userSearch"/>
   <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
  </beans:bean>

 </beans:beans>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

LDAP troubleshooting

No UserDetailsService registered error

*If you see an exception like this "**org.springframework.context.ApplicationContextException: No UserDetailsService registered**" you might add in your openkm.xml this line:*

```
<beans:bean id="userDetailService" class="com.openkm.spring.UserTenantDetailsService"
```

This issue only affects versions of OpenKM where the "remember me" feature was included. Take a look at the Changelog to check if your OpenKM is affected.

At Security configuration parameters you can find more information about "remember me" configuration.

javax.naming.PartialResultException error

*If you see an exception like this "**javax.naming.PartialResultException**" probably you can not use Active Directory basic configuration.*

javax.naming.PartialResultException: Unprocessed Continuation Reference(s); remaining name 'cn=users,dc=company,dc=com'

Solution:

Enable ldap referral property.

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
principal.ldap.referral	***String***	`follow`

Read these articles:

- *Referrals in the JNDI*
- *JNDI Implementor Guidelines for LDAP Service Providers*

Can't connect to LDAP server

If your LDAP server is configured under SSL then you should use ldaps://

Solution:

*Go to **Administration > Configuration parameters**:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change server connection URL to ldaps.

Field / Property	Type	Description
principal.ldap.server	**String**	`ldaps://192.168.0.6:389`

Slow login or not able to login

It may be a problem with LDAP DNS names resolution.

Solution:

To prevent this issue - specially on Windows OS Family - it's a good practice to add all subdomains in application server's hosts file (/etc/hosts for linux or C:/Windows/system32/driver/etc/host for windows).

Example based with active directory (LDAP) with distinguished base name ***dc=company,dc=com*** *and server at* ***SBSSERVER*** *domain server name.*

```
10.10.1.2 company.com
10.10.1.2 SBSSERVER SBSSERVER.company.com
10.10.1.2 Schema.Configuration.company.com
10.10.1.2 Configuration.company.com
10.10.1.2 DomainDnsZones.company.com
10.10.1.2 ForestDnsZones.company.com
```

More information at OpenKM forum.

Force all users to be members of ROLE_USER

In a huge user AD repositories, can not be practice for the AD administrator to adding a new group to all the users, in this case we can force across xml login configuration that any logged user be member of ROLE_USER. Must be added the xml tag ***<beans:property name="defaultRole" value="ROLE_USER" />****.*

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta

<security:authentication-manager alias="authenticationManager">
     <security:authentication-provider ref="ldapAuthProvider" />
  </security:authentication-manager>
```

```
<beans:bean id="contextSource" class="org.springframework.security.ldap.DefaultSpring
  <beans:constructor-arg value="ldap://192.168.xxx.xxx"/>
  <beans:property name="userDn" value="CN=connect,OU=OpenKM,DC=company,DC=com"/>
  <beans:property name="password" value="****"/>
  <beans:property name="baseEnvironmentProperties">
     <beans:map>
        <beans:entry>
          <beans:key>
            <beans:value>java.naming.referral</beans:value>
          </beans:key>
          <beans:value>follow</beans:value>
        </beans:entry>
      </beans:map>
   </beans:property>
</beans:bean>

<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentica
  <beans:constructor-arg>
    <beans:bean class="org.springframework.security.ldap.authentication.BindAuthentic
      <beans:constructor-arg ref="contextSource"/>
      <beans:property name="userSearch" ref="userSearch"/>
    </beans:bean>
  </beans:constructor-arg>
  <beans:constructor-arg name="authoritiesPopulator" ref="defaultLdapAuthoritiesPopul
</beans:bean>

<beans:bean id="userSearch" class="org.springframework.security.ldap.search.FilterBas
  <beans:constructor-arg index="0" value="DC=company,DC=com" />
  <beans:constructor-arg index="1" value="(&amp;(sAMAccountName={0})(|(memberOf=CN=RO
  <beans:constructor-arg index="2" ref="contextSource" />
  <beans:property name="searchSubtree" value="true" />
</beans:bean>

  <beans:bean id="defaultLdapAuthoritiesPopulator" class="org.springframework.securit
  <beans:constructor-arg ref="contextSource"/>
      <beans:constructor-arg value="DC=company,DC=com"/>
      <beans:property name="groupSearchFilter" value="member={0}"/>
      <beans:property name="groupRoleAttribute" value="cn"/>
      <beans:property name="searchSubtree" value="true" />
      <beans:property name="convertToUpperCase" value="false" />
      <beans:property name="rolePrefix" value="" />
      <beans:property name="defaultRole" value="ROLE_USER" />
 </beans:bean>

 <!--Needed for remember-me services -->
 <beans:bean id="userDetailService" class="org.springframework.security.ldap.userdeta
  <beans:constructor-arg ref="userSearch"/>
  <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator"/>
 </beans:bean>

</beans:beans>
```

org.springframework.beans.factory.parsing.BeanDefinitionParsingException

*If you see an error like this "**org.springframework.beans.factory.parsing.BeanDefinitionParsingException**" probably you have some spring wrong version set in the openkm.xml:*

org.springframework.beans.factory.parsing.BeanDefinitionParsingException: Configuration problem: You cannot use a spring-security-2.0.xsd or spring-security-3.0.xsd or spring-security-3.1.xsd schema with Spring Security 3.2. Please update your schema declarations to the 3.2 schema.
Offending resource: URL [file:/desarrollo/tomcat/tomcat-8.0.34-openkm-7.0/openkm.xml]

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
        at
    org.springframework.beans.factory.parsing.FailFastProblemReporter.fatal(FailFastProblemReporter.java:59)
```

Solution:

Edit the openkm.xml file and remove the version numbers:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta
```

to

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
             xmlns:security="http://www.springframework.org/schema/security"
             xmlns:task="http://www.springframework.org/schema/task"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.springframework.org/schema/beans
                                 http://www.springframework.org/schema/beans/spring-b
                                 http://www.springframework.org/schema/security
                                 http://www.springframework.org/schema/security/sprin
                                 http://www.springframework.org/schema/task
                                 http://www.springframework.org/schema/task/spring-ta
```

LDAP: error code 8 - BindSimple: Transport encryption required

The transport encryption required error usually happens when you are trying to connect with ldaps (with SSL encryption) and you have not configured the trust certificate into the JAVA certs storage.

*First, check you are using "**ldaps://**" - **ends with s** - in your **openkm.xml** file for the connection, when you get this error you sould use it.*

*The first step is to **get the certificate from the server** for it follow the steps below:*

```
$ openssl s_client -showcerts -connect  your.ldap.server.com:636
```

The output will contain a number of entries delimited with:

```
-----BEGIN CERTIFICATE-----
aklfhskfadljasdl1340234234ASDSDFSDFSDFSDFSD
....
-----END CERTIFICATE-----
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Copy the last certificate entry into a file (ldapca.crt)

*Then, **add it to the java** keystore in $JRE_HOME/lib/security*

```
$ cd $JRE_HOME/lib/security

$ keytool -import -alias ldapca_self_sign -keystore cacerts -storepass changeit -file ldapca.crt
```

*Finally **enable trust store** into the file $TOMCAT_HOME/bin/setenv.sh*

```
# JAVA enable trust store
JAVA_OPTS="$JAVA_OPTS -Djavax.net.ssl.trustStore=/etc/pki/java/cacerts -Djavax.net.ss
```

Additional information:

- *https://docs.spring.io/spring-ldap/docs/1.2.0/reference/#dir-context-url*
- *https://stackoverflow.com/questions/9903560/spring-security-ssl-ldap-and-no-certificate*

Ldap error: Caused by java.security.cert.CertificateException: No subject alternative DNS name matching

In the log maybe show some stack trace error like:

```
Caused by: java.security.cert.CertificateException: No subject alternative DNS name m
        at sun.security.util.HostnameChecker.matchDNS(HostnameChecker.java:214)
        at sun.security.util.HostnameChecker.match(HostnameChecker.java:96)
        at sun.security.ssl.X509TrustManagerImpl.checkIdentity(X509TrustManagerImpl.j
        at sun.security.ssl.X509TrustManagerImpl.checkIdentity(X509TrustManagerImpl.j
        at sun.security.ssl.X509TrustManagerImpl.checkTrusted(X509TrustManagerImpl.ja
        at sun.security.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerI
        at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:
        ... 72 more
```

Or like:

```
Caused by: org.springframework.ldap.CommunicationException: simple bind failed: ldap.
        at org.springframework.ldap.support.LdapUtils.convertLdapException(LdapUtils.
        at org.springframework.ldap.core.support.AbstractContextSource.createContext(
        at org.springframework.ldap.core.support.AbstractContextSource.doGetContext(A
        at org.springframework.ldap.core.support.AbstractContextSource.getReadOnlyCon
        at org.springframework.ldap.core.LdapTemplate.executeReadOnly(LdapTemplate.ja
        at org.springframework.security.ldap.SpringSecurityLdapTemplate.searchForSing
        at org.springframework.security.ldap.search.FilterBasedLdapUserSearch.searchF
        at org.springframework.security.ldap.authentication.BindAuthenticator.authent
        at org.springframework.security.ldap.authentication.LdapAuthenticationProvide
```

Modify JVM configuration parameters in setenv.sh or setenv.bat adding the JVM configuration parameter:

```
-Dcom.sun.jndi.ldap.object.disableEndpointIdentification=true
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Log Configuration

Logback based Log Configuration

Available since 6.4.28 version.

OpenKM uses logback which is a Java-based loggin utility. This is useful, for example when you need to get more information about something that is not working properly in OpenKM, for example third-party application configuration or Spring authentication. By default only log messages with INFO level are shown, but you can increase this log information by raising the log level to WARN or DEBUG. The log configuration file is ***$TOMCAT_HOME/logbackConfig.xml*** *here you can change the error level message generated by a give class or package.*

When OpenKM detect a modification in the ***logbackConfig.xml*** *file, it will reload it so you don't need to stop and start OpenKM to make these changes effectives.*

Third-party application execution

This is useful when something like a document preview fails, because this way OpenKM will log every external program execution:

```
<logger name="com.openkm.util" level="DEBUG" />
```

Enable log authentication

If you need to debug the security configuration, edit the ***logbackConfig.xml*** *file and add this line:*

```
<logger name="org.springframework.security" level="DEBUG" />
```

Log4j based Log Configuration

Deprecated since 6.4.28 version.

OpenKM uses log4j which is a Java-based loggin utility. This is useful, for example when you need to get more information about something that is not working properly in OpenKM, for example third-party application configuration or Spring authentication. By default only log messages with INFO level are shown, but you can increase this log information by raising the log level to WARN or DEBUG. The log configuration file is ***$TOMCAT_HOME/conf/log4j.properties*** *here you can change the error level message generated by a give class or package.*

When OpenKM detect a modification in the ***log4j.properties*** *file, it will reload it so you don't need to stop and*

start OpenKM to make these changes effectives.

Third-party application execution

This is useful when something like a document preview fails, because this way OpenKM will log every external program execution:

```
log4j.logger.com.openkm.util.ExecutionUtils=DEBUG
```

Enable log authentication

If you need to debug the security configuration, edit the ***log4j.properties*** *file and add this line:*

```
log4j.logger.org.springframework.security=DEBUG
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OpenOffice and LibreOffice configuration

> *After changing **system.openoffice.path** parameter is required to restart application.*

Linux

Debian & Ubuntu

> *OpenKM has been roughly tested with recent build of **OpenOffice** and **LibreOffice**, but in recent Linux distros only **LibreOffice** is available to install. Anyway it works pretty well.*

Install application:

```
$ apt-get install libreoffice
```

Install languages:

```
$ apt-get install libreoffice-l10n-en-gb
```

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
system.openoffice.path	***String***	*The path may change depending your OS version. To verify that your path is correct, take a look into your chosen path and it should be a folder named program, otherwise the path is not correct.* `/usr/bin/libreoffice`

CentOS

Install the application:

```
$ yum install libreoffice
```

Install languages:

```
$ yum install libreoffice-langpack-en
```

> *By default, OpenOffice or LibreOffice is not present on Red Hat repository. If you're using this OS then you should try to install manually.*

Download and install one of them from:

- *LibreOffice: http://www.libreoffice.org*
- *OpenOffice: http://www.openoffice.org*

Check if you've installed all needed packages:

```
$ rpm -qa | grep openoffice
```

These packages are mandatory:

- ***openoffice.org-ure-3.1.1-19.5.el5_5.6*** *(ure)*
- ***openoffice.org-headless-3.1.1-19.5.el5_5.6*** *(headless)*
- ***openoffice.org-pyuno-3.1.1-19.5.el5_5.6*** *(pyuno)*

The name of the packages can change depending your OpenOffice version. Take these as reference for your check.

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
system.openoffice.path	***String***	*The path can change depending your OS version. To verify your path is correct, into your chosen path should be a folder named program, otherwise the path is not correct.* `/usr/bin/libreoffice`

Windows

Download and install one of them:

- *LibreOffice: http://www.libreoffice.org*
- *OpenOffice: http://www.openoffice.org*

Our suggestion is to use OpenOffice in Windows OS. In some Windows OS we've found unexpected problems that are automatically solved changing LibreOffice to OpenOffice. Take it only as a suggestion.

*Go to **Administration > Configuration parameters**:*

Field / Property	*Type*	*Description*
		The path can change depending your OS version. To verify your path

system.openoffice.path	***String***	*is correct, into your chosen path should be a folder named program, otherwise the path is not correct.* `c:\Program files\OpenOffice 4`

Check configuration

- *Go to Administration > Configuration parameters and click on check icon . If all is right you should see under* ***system.openoffice.path*** *parameter an* ***OK*** *status.*
- *After application starts you should see a process called* ***soffice.bin****.*
- *Upload a MS Office or OpenOffice file. Select the document and from toolbar choose the option* ***Download as PDF*** *.*
- *Take a look at $TOMCAT_HOME/logs/catalina.log file. At the end of the startup process you should see something like this:*

```
2015-03-17 17:10:45,792 [pool-2-thread-1] INFO  com.openkm.servlet.RepositoryStartupS
2015-03-17 17:10:45,813 [pool-2-thread-1] INFO  com.openkm.util.DocConverter - *** Bu
2015-03-17 17:10:45,813 [pool-2-thread-1] INFO  com.openkm.util.DocConverter - system
2015-03-17 17:10:45,813 [pool-2-thread-1] INFO  com.openkm.util.DocConverter - system
2015-03-17 17:10:45,813 [pool-2-thread-1] INFO  com.openkm.util.DocConverter - system
2015-03-17 17:10:45,866 [pool-2-thread-1] INFO  org.artofsolving.jodconverter.office.
2015-03-17 17:10:45,915 [OfficeProcessThread-0] WARN  org.artofsolving.jodconverter.o
2015-03-17 17:10:45,934 [OfficeProcessThread-0] INFO  org.artofsolving.jodconverter.o
2015-03-17 17:10:45,956 [OfficeProcessThread-0] INFO  org.artofsolving.jodconverter.o
2015-03-17 17:10:48,818 [OfficeProcessThread-0] WARN  org.artofsolving.jodconverter.o
2015-03-17 17:10:48,825 [OfficeProcessThread-0] INFO  org.artofsolving.jodconverter.o
2015-03-17 17:10:48,832 [OfficeProcessThread-0] INFO  org.artofsolving.jodconverter.o
```

Read OpenOffice LibreOffice configuration parameters to get more information about all available OpenOffice & LibreOffice configuration parameters.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Security configuration

OpenKM can work with several security managers, depending on the selected value, the manner in which the result is evaluated is different. This application has separated managers, one for the search engine and other for the rest of the application.

Search engine managers:

- *lucene.*
- *am_more, am_window or am_limited.*

> *The fastest one is **lucene**, because it is the Lucene search engine which restricts the search results based on the node security. The others are **am_more**, **am_window** and **am_limited**, which make use of the AccessManager - security.access.manager value - to restrict the search results, these will reduce OpenKM performance (at least in search) but will match customer requirements. The performance can be improved by customizing **com.openkm.cache.parentNodePermissions** cache.*

Rest of application manager:

> *Access manager works with plugin architecture. From*

Type	*Name*	*Description*
Default	***com.openkm.plugin.access.DbSimpleAccessManager***	*Default OpenKM security manager. Analyze only the current node, not all the hierarchy.*
Recursive	***com.openkm.plugin.access.DbRecursiveAccessManager***	*Analyze all the hierarchy.* > *Analyze all the hierarchy will take more time and may cause performance issues.*
Read recursive	***com.openkm.plugin.access.DbReadRecursiveAccessManager***	*Analyze all the hierarchy (only for read access).* > *Analyze all the hierarchy will take more time and may*

		cause performance issues.
Default file plan	***com.openkm.plugin.access.DbFilePlanAccessManagerWrapper***	*Analyze based on document type and file plan restrictions.* *Only should be enabled when file plan is enabled.*

*The **Deafult** option only evaluates the selected node. With **recursive**, the security is evaluated starting with the selected node until the repository root (so, evaluating form bottom to top). The **"Read recursive"** is used to prevent showing documents in search results when any folder in the path doesn't have read access for the user who ran the query. Should be used in combination with **am_more**.*

Application by default is configured with:

Field / Property	*Type*	*Description*
security.access.manager	***String***	`com.openkm.plugin.access.DbSimpleAccessManager`
security.search.evaluation	***String***	`lucene`

Diferences betweem simple and recursive

- *okm:root*
  - *level-1a*
    - *level-2a*
      - *level-3a*
        - *alpha.doc*
        - *beta.doc*
        - *gamma.doc*
      - *level-3b*
  - *level-1b*
  - *level-1c*

***When using simple** if you revoke permission from the level-2a folder and the user tries to view the level-3a folder, OpenKM will still show the documents to this user. When the user performs a search, OpenKM will return these documents as search results. However, the user can't access the documents because when OpenKM tries to access the document, it will open every path*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

location and will fail to open level-2a because the user has no rights. But these document can be accessed using the API. You need to apply revoke recursively to prevent the user from seeing these documents, so they won't be shown in the search results.

***When using recursive** if you revoke permission from the level-2a folder and the user tries to view the level-3a folder, OpenKM will report an error because Access Manager also evaluates the permission of the level-2a ancestor node. When the user performs a search, OpenKM won't show these documents in the search results. Keep in mind that this Access Manager implementation is slower because it needs to evaluate more nodes: the more depth node the more parent nodes need to be evaluated.*

Extending security

*You can use the extend security to download files, start workflows, add, remove or modify property groups (metadata) or compact history. To enable extended security evaluation you must change the value of **security.extended.mask** parameter.*

After the value is changed, refresh the browser to take effect in user view.

Values:

- *DOWNLOAD = 1024;*
- *START_WORKFLOW = 2048;*
- *COMPACT_HISTORY = 4096;*
- *PROPERTY_GROUP = 8192;*

For example, to enable download and property groups you should put the mask 9216 (1024 + 8192)

Field / Property	*Type*	*Description*
security.extended.mask	***Integer***	`9216`

Configuration for security changes which take a long time

*By default when you change the security of a node in standard view, every time you click to grant or revoke a permission the action is performed by OpenKM. This means that if you want to change the security of 10 users, 10 commands are sent to the server. This became a problem when every change affects several hundred (or even thousands) of nodes. There's an alternative security configuration parameter named **security.mode.multiple** which enables multiple changes at a time, so it's so faster.*

Field / Property	*Type*	*Description*
security.mode.multiple	***Boolean***	`true`

Some security changes can take a long time to be completed because affect a lot of nodes. It's a good practice to do it as a

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

background task. For it there's a configuration parameter named ***security.live.change.node.limit*** *used to set the limit of nodes which can be changed in "live" or "realtime" mode. By default this value is set to 100, which means that any change which affect to more than 100 nodes will be automatically done as a background task.*

Field / Property	*Type*	*Description*
security.live.change.node.limit	***Integer***	100

To check the progress of these pending tasks go to ***Administration > Statistics > Pending task queue***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

SSO Configuration

> ***Before starting the SSO configuration you should have well configured for AD.***
>
> *Take a look at LDAP configuration.*

The aim of this configuration is to allow users to login into OpenKM with their Windows credentials. To do it, it is necesary to configure some stuff, including client's browsers.

In our case the whole system is integrated by:

- *Active Directory in Windows 2008 and Windows 2012 server*
- *Tomcat server in a Linux machine*
- *Client on Windows 7 (Browsers Internet Explorer and Firefox)*

The following guides have been used as a start point:

- *http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=spnego-spring*
- *http://wiki.openkm.com/index.php/Active_Directory_-_OpenKM_6.2*
- *LDAP configuration parameters*
- *http://projects.spring.io/spring-security-kerberos/*
- *http://docs.spring.io/spring-security-kerberos/docs/1.0.0.RC2/reference/htmlsingle/*

Active Directory

It is supposed that there is an Active Directory server installed and running (LDAP implementation) where it is going to be configured all the user accounts that will be used later.

The steps to follow are:

1. *Create a Computer object inside Active Directory that will correspond to the Linux server*
    - *Once done, modify its properties and activate the following option:*
        - ***Trusted for delegation**: This step is important because, if not enabled, negotiation between clients and servers won't be possible.*
2. *The server must authenticate against Active Directory before validating the tickets sent by the client. This authentication is performed using a file (keytab) that map the service (application server) with a domain user.* ***Create a domain user**. **This user must be used only and exclusively for this purpose because if this user is used to log into another computer, it won't be able to be used to log on the application server**. Once created the user, its properties are modified to not allow the modification of the password and the non expiration of the account.*
3. ***Generate keytab file**. This file is an encoded copy of the private key of an user in Kerberos. It is used by the application*

web as an authentication proxy of the credentials sent by the user. The following command is executed:

```
C:\>  ktpass -princ HTTP/tomcat.example.com@EXAMPLE.COM -mapuser user@EXAMAPLE.COM -crypto all -kvno 0 -ptype KRB5_NT_PRINCIPAL -pass * -out web.keytab
```

It is important to highlight :

- *1. **tomcat.example.com@EXAMPLE.COM**: Full name of the Tomcat server and its domain.*
- *2. **user@EXAMAPLE.COM**: The user created previously.*
- *3. **web.keytab**: The file where the private key that will be used by the Tomcat server is exported.*

4. *Register new SPN services. A Service Principal Name (SPN) needs to be setup with HTTP and a server name: in example: tomcat.example.org where Tomcat is running. This is used with domain user and its keytab is then used as a service credential:*

```
C:\> setspn -A HTTP/tomcat.example.org@EXAMPLE.ORG user
C:\> setspn -A HTTP/tomcat.example.org user
```

> *Where tomcat.example.org is the machine where tomcat servlet container is running and user is the user previously created.*
>
> *Usually the previous command "ktpass" should create the service HTTP/tomcat.example.org, when it's created will be raised an error indicating is yet present.*

5. *Verify services has been correctly created and linked to the user:*

```
C:\> setspn -l user
```

6. *Finally create the groups that will be assigned to the users and which the corresponding roles will be obtainined from (You should have it from previous AD integration step). It is recomended to create:*
   - *An Organization Unit (OU) with name OpenKM*
   - *Inside the previous Organization Unit , create the groups that will be used as roles*
     - *ROLE_USER*
     - *ROLE_ADMIN*

> *If you want to create the spn services again, the best way is deleting the user and create again in th domain. Also there's available the command:*
>
> ```
> $ setspn -D openkmsso
> ```
>
> *But we do not suggest it. It's always better deleting the user.*

Linux Machine

The first thing to do is to include the application server inside the domain, if it was not done previously. To do it follow the guide:

http://cerowarnings.blogspot.com.es/2011/11/how-to-linux-en-dominio-windows.html

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

One thing to consider is to add the following configuration to the file /etc/host due to previously known problems:

```
# ldap in /etc/hosts
192.168.1.110 example.com
192.168.1.110 Schema.Configuration.example.com
192.168.1.110 Configuration.example.com
192.168.1.110 DomainDnsZones.example.com
192.168.1.110 ForestDnsZones.example.com
```

Kerberos configuration file

Install kerberos client:

```
$ apt-get krb5-user krb5-config
```

Before starting the application Kerberos must be configured properly to validate tickets correctly. In Linux systems this file is located at "/etc/krb5.conf".

In this file Domain, KDC server and allowed algorithms must be specified. The most importans algorithms are ***arcfour-hmac-md5*** *and* ***rc4-hmac*** *which Microsoft Active Directory offers.*

Preservate almost krb5.con default configuration and make only the suggested changed shown below.

```
[libdefaults]
default_realm = EXAMPLE.COM
#these are mandatory for kerberos.
permitted_enctypes = aes128-cts rc4-hmac
default_tgs_enctypes = aes128-cts rc4-hmac
default_tkt_enctypes = aes128-cts rc4-hmac

[realms]
EXAMPLE.COM = {
  kdc = SERVER.example.com
  admin_server = SERVER.example.com
  default_domain = EXAMPLE.COM
}

[domain_realm]
example.com = EXAMPLE.COM
.example.com = EXAMPLE.COM
```

Configuration files

The appContext.xml file might be modified and the ***security:http*** *entry do not be exactly as is shown below (some addional atrributtes).*

To change the authentication type to use it is necessary to modify the following two files:

- *$CATALINA_HOME/webapps/OpenKM/WEB-INF/appContext.xml:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change the following line:

```
<security:http access-decision-manager-ref="accessDecisionManager" access-denied-page
```

to

```
<security:http access-decision-manager-ref="accessDecisionManager" access-denied-page
```

Under this line add:

```
<security:custom-filter ref="spnegoAuthenticationProcessingFilter" position="BASIC_AU
```

A detail to take into account inside this file, is that if it is required to filter the access not only but the authentication but the roles too, it is needed to replace the attribute IS_AUTHENTICATED_FULLY to an authorize role list separated by comas (ROLE_USER, ROLE_ADMIN)

- *$CATALINA_HOME/openkm.xml*

  Comment the line:

```
 <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider>
      <security:password-encoder hash="md5"/>
      <security:jdbc-user-service
        data-source-ref="dataSource"
        users-by-username-query="select usr_id, usr_password, 1 from OKM_USER where u
        authorities-by-username-query="select ur_user, ur_role from OKM_USER_ROLE whe
    </security:authentication-provider>
 </security:authentication-manager>
```

And add the following:

```
<beans:bean id="spnegoEntryPoint" class="org.springframework.security.kerberos.web.au
    <beans:constructor-arg value="/login" />
</beans:bean>

<beans:bean id="kerberosAuthenticationProvider" class="org.springframework.security.k
    <beans:property name="userDetailsService" ref="ldapUserDetailsService" />
    <beans:property name="kerberosClient">
        <beans:bean class="org.springframework.security.kerberos.authentication.sun.S
            <beans:property name="debug" value="true"/>
        </beans:bean>
    </beans:property>
</beans:bean>

<!--beans:bean id="kerberosServiceAuthenticationProvider" class="org.springframework.
<beans:bean id="kerberosServiceAuthenticationProvider" class="com.openkm.spring.OKMKe
    <beans:property name="ticketValidator">
        <beans:bean class="org.springframework.security.kerberos.authentication.sun.S
            <beans:property name="servicePrincipal" value="HTTP/tomcat.example.com@EX
            <beans:property name="keyTabLocation" value="file:/path/to/web.keytab" />
            <beans:property name="debug" value="true" />
```

```
        </beans:bean>
    </beans:property>
    <beans:property name="userDetailsService" ref="ldapUserDetailsService" />
</beans:bean>

<!-- Ticket security manager -->
<security:authentication-manager id="authenticationManagerSpnego">
    <security:authentication-provider ref="kerberosAuthenticationProvider" />
    <security:authentication-provider ref="kerberosServiceAuthenticationProvider" />
</security:authentication-manager>

<beans:bean id="spnegoAuthenticationProcessingFilter" class="org.springframework.secu
    <beans:property name="authenticationManager" ref="authenticationManagerSpnego" />
</beans:bean>

<beans:bean id="authorizationContextSource" class="org.springframework.security.kerbe
    <beans:constructor-arg value="ldap://server.example.com:389/dc=example,dc=com" />
    <beans:property name="loginConfig" ref="LoginConfig"/>
</beans:bean>

<beans:bean id="LoginConfig" class="org.springframework.security.kerberos.client.conf
    <beans:property name="servicePrincipal" value="HTTP/tomcat.example.com@EXAMPLE.CO
    <beans:property name="keyTabLocation" value="file:/path/to/web.keytab" />
    <beans:property name="useTicketCache" value="false" />
    <beans:property name="isInitiator" value="true" />
    <beans:property name="debug" value="true" />
</beans:bean>

<beans:bean id="ldapUserDetailsService" class="org.springframework.security.ldap.user
    <beans:constructor-arg ref="FilterBasedLdapUserSearch" />
    <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator" />
    <beans:property name="UserDetailsMapper" ref="UserDetailsMapper" />
</beans:bean>

<beans:bean id="FilterBasedLdapUserSearch" class="org.springframework.security.ldap.s
    <beans:constructor-arg value="" />
    <!--beans:constructor-arg value="dc=example,dc=com" /-->
    <beans:constructor-arg value="(| (userPrincipalName={0}) (sAMAccountName={0}))" /
    <beans:constructor-arg ref="authorizationContextSource" />
</beans:bean>

<beans:bean id="defaultLdapAuthoritiesPopulator" class="com.openkm.spring.OKMLdapAuth
<beans:bean id="UserDetailsMapper" class="org.springframework.security.ldap.userdetai

<beans:bean id="ldapAuthProvider" class="org.springframework.security.ldap.authentica
    <beans:constructor-arg>
        <beans:bean class="org.springframework.security.ldap.authentication.BindAuthe
            <beans:constructor-arg ref="authorizationContextSource"/>
            <beans:property name="userSearch" ref="FilterBasedLdapUserSearch"/>
        </beans:bean>
    </beans:constructor-arg>
    <beans:constructor-arg ref="defaultLdapAuthoritiesPopulator" />
</beans:bean>

<!-- Alternative Ldap security manager -->
<security:authentication-manager alias="authenticationManager">
    <security:authentication-provider ref="ldapAuthProvider" />
</security:authentication-manager>
```

The important configuration data to take into account is:

- *ldap://server.example.com:389/dc=example,dc=com" that is the location of the ldap server.*
- *servicePrincipal that is the service previusly created at the server: HTTP/tomcat.example.org@EXAMPLE.COM*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *keyTabLocation: that is the location inside the file system of the linux machine of the keytab file created in the Active directory server. (It is possible to use classpath: but is preferable use file:)*

Configure LDAP in OpenKM

Once done the previously configurations, the Tomcat server can be start up. To allow user name and role to be available inside OpenKM, it is necessary to configurate LDAP. To do it, it's recomended to follow the steps of LDAP configuration parameters.

Client browsers:

To allow the automatic credentials negotiation, it is also necessary to configure the client web browsers. This is detailed at:

http://docs.spring.io/spring-security-kerberos/docs/1.0.0.RC2/reference/htmlsingle/#browserspnegoconfig

- *Firefox*
    1. *Open Firefox.*
    2. *At address field, type **about:config**.*
    3. *In filter/search, type **negotiate**.*
    4. *Parameter **network.negotiate-auth.trusted-uris** may be set to default https:// which doesn't work for you. Replace this with your server address (tomcat.example.com in our example).*

- *Internet Explorer:*
    1. *Open Internet Explorer.*
    2. *Click Tools > Intenet Options > Security tab.*
    3. *In Local intranet section make sure your server is trusted by i.e. adding it into a list (tomcat.example.com in our example)..*
    4. *In advanced options: check the option "Enable Integrated Windows Authentication" is enabled*
    5. *Add your server address (tomcat.example.com in our example)*
- *Chrome:*
    1. *Open Chrome.*
    2. *Click Configuration> Show Advanced Configuration > Change proxy configuration > Security tab.*
    3. *In Local intranet section make sure your server is trusted by i.e. adding it into a list (tomcat.example.com in our example)..*
    4. *In advanced options: check the option "Enable Integrated Windows Authentication" is enabled.*
    5. *Add your server address (tomcat.example.com in our example)*

Throubleshootings

SSO configuration is quite complex. Below you will find some clues and tools what can be useful while trying to configure it.

Enable log for SSO authentication

Modify the log configuration file and add the line:

```
<logger name="org.springframework.security" level="DEBUG" />
```

More information about log configuration at Log configuration.

Check kerberos authentication from linux command line

Execute the command (it's case sensitive and uppercase is important) :

```
$ kinit user@EXAMPLE.ORG
```

If all is right, you will be inquired for setting the password and the login from linux is make. If it's going wrong you have something wrong in your kerberos linux client configuration (krb5.conf).

Listing keys

Almost problems we have found, comes with bad we.keytab creation file. If you get errors like:

```
rg.springframework.ldap.AuthenticationException: GSSAPI; nested exception is javax.na
 at org.springframework.ldap.support.LdapUtils.convertLdapException(LdapUtils.java:18
 at org.springframework.ldap.core.support.AbstractContextSource.createContext(Abstrac
 at org.springframework.ldap.core.support.AbstractContextSource.doGetContext(Abstract
 at org.springframework.ldap.core.support.AbstractContextSource.getReadOnlyContext(Ab
 at org.springframework.ldap.core.LdapTemplate.executeReadOnly(LdapTemplate.java:791)
 at org.springframework.security.ldap.SpringSecurityLdapTemplate.searchForSingleEntry
 at org.springframework.security.ldap.search.FilterBasedLdapUserSearch.searchForUser(
 at org.springframework.security.ldap.userdetails.LdapUserDetailsService.loadUserByUs
 at com.openkm.spring.OKMKerberosServiceAuthenticationProvider.authenticate(OKMKerber
 at org.springframework.security.authentication.ProviderManager.authenticate(Provider
 at org.springframework.security.kerberos.web.authentication.SpnegoAuthenticationProc
 at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(Fil
 at org.springframework.security.web.authentication.AbstractAuthenticationProcessingF
 at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(Fil
 at org.springframework.security.web.authentication.logout.LogoutFilter.doFilter(Logo
 at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(Fil
 at org.springframework.security.web.context.request.async.WebAsyncManagerIntegration
 at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter
 at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(Fil
 at org.springframework.security.web.context.SecurityContextPersistenceFilter.doFilte
 at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(Fil
 at org.springframework.security.web.FilterChainProxy.doFilterInternal(FilterChainPro
 at org.springframework.security.web.FilterChainProxy.doFilter(FilterChainProxy.java:
 at org.springframework.web.filter.DelegatingFilterProxy.invokeDelegate(DelegatingFil
 at org.springframework.web.filter.DelegatingFilterProxy.doFilter(DelegatingFilterPro
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:22
 at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:16
 at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java
 at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:168)
 at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:98)
```

```
  at org.apache.catalina.valves.AccessLogValve.invoke(AccessLogValve.java:927)
  at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:118)
  at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:407)
  at org.apache.coyote.http11.AbstractHttp11Processor.process(AbstractHttp11Processor.
  at org.apache.coyote.AbstractProtocol$AbstractConnectionHandler.process(AbstractProt
  at org.apache.tomcat.util.net.JIoEndpoint$SocketProcessor.run(JIoEndpoint.java:309)
  at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
  at java.lang.Thread.run(Thread.java:745)
Caused by: javax.naming.AuthenticationException: GSSAPI [Root exception is javax.secu
  at com.sun.jndi.ldap.sasl.LdapSasl.saslBind(LdapSasl.java:168)
  at com.sun.jndi.ldap.LdapClient.authenticate(LdapClient.java:235)
  at com.sun.jndi.ldap.LdapCtx.connect(LdapCtx.java:2740)
  at com.sun.jndi.ldap.LdapCtx.<init>(LdapCtx.java:316)
  at com.sun.jndi.ldap.LdapCtxFactory.getUsingURL(LdapCtxFactory.java:193)
  at com.sun.jndi.ldap.LdapCtxFactory.getUsingURLs(LdapCtxFactory.java:211)
  at com.sun.jndi.ldap.LdapCtxFactory.getLdapCtxInstance(LdapCtxFactory.java:154)
  at com.sun.jndi.ldap.LdapCtxFactory.getInitialContext(LdapCtxFactory.java:84)
  at javax.naming.spi.NamingManager.getInitialContext(NamingManager.java:684)
  at javax.naming.InitialContext.getDefaultInitCtx(InitialContext.java:307)
  at javax.naming.InitialContext.init(InitialContext.java:242)
  at javax.naming.ldap.InitialLdapContext.<init>(InitialLdapContext.java:153)
  at org.springframework.ldap.core.support.LdapContextSource.getDirContextInstance(Lda
  at org.springframework.security.kerberos.client.KerberosLdapContextSource.access$001
  at org.springframework.security.kerberos.client.KerberosLdapContextSource$1.run(Kerb
  at org.springframework.security.kerberos.client.KerberosLdapContextSource$1.run(Kerb
  at java.security.AccessController.doPrivileged(Native Method)
  at javax.security.auth.Subject.doAs(Subject.java:356)
  at org.springframework.security.kerberos.client.KerberosLdapContextSource.getDirCont
  at org.springframework.ldap.core.support.AbstractContextSource.createContext(Abstrac
  ... 39 more
Caused by: javax.security.sasl.SaslException: GSS initiate failed [Caused by GSSExcep
  at com.sun.security.sasl.gsskerb.GssKrb5Client.evaluateChallenge(GssKrb5Client.java:
  at com.sun.jndi.ldap.sasl.LdapSasl.saslBind(LdapSasl.java:123)
  ... 58 more
Caused by: GSSException: No valid credentials provided (Mechanism level: Server not f
  at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:710)
  at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:248)
  at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:179)
  at com.sun.security.sasl.gsskerb.GssKrb5Client.evaluateChallenge(GssKrb5Client.java:
  ... 59 more
Caused by: KrbException: Server not found in Kerberos database (7)
  at sun.security.krb5.KrbTgsRep.<init>(KrbTgsRep.java:70)
  at sun.security.krb5.KrbTgsReq.getReply(KrbTgsReq.java:192)
  at sun.security.krb5.KrbTgsReq.sendAndGetCreds(KrbTgsReq.java:203)
  at sun.security.krb5.internal.CredentialsUtil.serviceCreds(CredentialsUtil.java:309)
  at sun.security.krb5.internal.CredentialsUtil.acquireServiceCreds(CredentialsUtil.ja
  at sun.security.krb5.Credentials.acquireServiceCreds(Credentials.java:456)
  at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:641)
  ... 62 more
Caused by: KrbException: Identifier doesn't match expected value (906)
  at sun.security.krb5.internal.KDCRep.init(KDCRep.java:143)
  at sun.security.krb5.internal.TGSRep.init(TGSRep.java:66)
  at sun.security.krb5.internal.TGSRep.<init>(TGSRep.java:61)
  at sun.security.krb5.KrbTgsRep.<init>(KrbTgsRep.java:55)
  ... 68 more
```

Probably your web.keytab is wrong. ***Ensure you have executed "ktpass" command before executing the "setspn" ones.***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Document expiration is a notification system alert which can be used to advice users nearly document expiration and expired documents.

*This option is only available if you have enabled the **Document expiration** feature.*

Document expiration is based on:

- *Metadata group feature.*
- *Crontab feature.*
- *Report feature.*

More information at Document expiration extension.

Configuration

Download the expiration files document_expiration_7.zip.

Register parameters

*Execute SQL from **Administration** > **Tools** > Database query.*

```
-- NOTIFICATION PARAMETERS
DELETE FROM OKM_CONFIG WHERE CFG_KEY LIKE '%expiration.%';
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.date.patter
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.expiration.
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.notificatio
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.notificatio
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.alert.subje
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.alert.templ
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.expired.sub
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('expiration.expired.tem
```

More information at Database query.

Configure parameters

Field / Property	Type	Description
expiration.date.pattern	**String**	*Set the date pattern.* `dd-MM-yyyy HH:mm:ss`
expiration.notification.subject	**String**	*Mail subject.* `New uploaded document with expiration`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

expiration.notification.template	***String***	*Mail body.* `New document uploaded ${docUrl}` ` Expiration date:${date} `
expiration.expiration.alert.days	***Integer***	*Set the number of days before expiration when it will be send the expiration alert message.* `15` *Default document expiration set an unique value for all document. This is the default logic but can be implemented your own.*
expiration.alert.subject	***String***	*Mail subject.* `Alert documents will expire`
expiration.alert.template	***String***	*Mail body.* `Document ${docUrl} will expire at ${date} `

These parameters are set at Administration > Configuration parameters view.

Register metadata group

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.7//EN" "http://www
<property-groups>
  <property-group label="Expiration" name="okg:expiration">
    <input label="Expiration date" name="okp:expiration.date" type="date">
      <validator type="req"/>
    </input>
    <select label="State" name="okp:expiration.state" type="simple">
      <option label="Notify" value="notify" selected="true"/>
      <option label="Valid" value="valid"/>
      <option label="Expired" value="expired"/>
      <validator type="req"/>
    </select>
    <select label="Notify" name="okp:expiration.groupnotify" type="multiple" classNam
      <validator type="req"/>
    </select>
    <select label="Notify expiration alert" name="okp:expiration.groupalert" type="mu
      <validator type="req"/>
    </select>
    <select label="Notify expirated" name="okp:expiration.groupexpired" type="multipl
      <validator type="req"/>
    </select>
  </property-group>
```

```
</property-groups>
```

Crontab task

- *Copy jar files into the $TOMCAT_HOME/plugins folder.*
- *Restart OpenKM (that will register crontab tasks).*

Task	Expression	Description
InitialNotificationation.jar	***0 */1 * * * ***	*After is added metadata group to the node is sent the "initial notification" mail.* *The crontab task is executed each minute looking for new nodes with metadata group.*
AlertExpirationNotificationation.jar	***0 30 0 * * ***	*Sent daily mail alert for documents expire soon.*
ExpirationNotificationation.jar	***0 0 0 * * **	*Sent daily mail alert for documents expired today.* *This crontab task should be executed before **AlertExpirationNotificationation** task.*

More information at Crontab.

Register reports

Report	Description
ValidDocuments.jrxml	*List all valid documents.*
ExpiredDocuments.jrxml	*List all expired documents.*
NextExpirationDocuments.jrxml	*List documents what will expire soon.*

More information at Reports.

See also Profiles for configuring reports in user interface menu view.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Two Factor Authentication

Two factor authentication (2FA) provides extra security to OpenKM login. In addition to the typical login and password, once the user is authenticated, will be asked for an extra six digits code generate in a mobile application called Google Authenticator.

Prerequisite

Because two factor authentication has been implemented using Google Authenticator, it is necessary to download and install the application from Google Play or Apple market.

> *https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2*

Once installed and configured, this application does not event need Internet connection.

OpenKM configuration

Each user can configure his own account to use 2FA. To do it, he has to go to ***Tools*** > ***Preferences*** > ***User Configuration***.

If 2FA was enabled by the administrator, at the bottom of the User Configuration dialog you would be able to see a new option ***2FA***. *Depending on whether 2FA is enabled ypu would see:*

- *Disable button: is 2FA was already configured for this user.*
- *Enable button: is 2FA has not been configured yet.*

If you click on enable button, a new window with the following parameters will appear:

- *The current user id to be configured.*
- *A QR code which you should scan from Google Authenticator application.*
- *A text box where you have to write the verification code generated by Google Authenticator (once QR code has been scanned).*

Once you have complete these steps, please click on the accept button.

Scan QR code

To scan the QR code, open Google Authenticator on your mobile and perfom the following steps:

1. *Click on configure account.*
2. *Select* ***Scan code***.
3. *Focus the mobile toward the the QR image of the screen.*
4. *Once the image is properly scanned the accound for OpenKM is added.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

5. *Now you will see a square with a number (code) which you will have to introduce when required.*

These generated codes are time based: they will expiry every 60 seconds. If you pay attention to Google Authenticator screen, you will see how the code changes every period of time.

Google Authenticator use

*Once **2FA** is configured on your OpenKM account, next time you log into OpenKM, after your login and password are verified, a new screen will appear, where it will be required to fill in, a code that it will have to be obtained from the application previously configured inside Google Authenticator application.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring CIFS

The aim of this configuration is to allow to see OpenKM as a remote drive using ***CIFS*** *protocol. Bellow it will be depicted two ways of configuiring it:*

- *Authentication using Database*
- *Authentication using Active Directory*

Database Authentication

This is the simplest case. You have to configure these parts:

- *OpenKM Server*
- *OpenKM Configuration*
- *Windows Client*

OpenKM server

If the system could not be started with root permissions the ports commented below, must be over 1024 (the default system ports are 138,137,139 and 445), so it is necessary to redirect the original ports to the new ones. To do this it is necessary to execute the following commands as ***root or using sudo****:*

```
modprobe iptable_nat
iptables -F
iptables -t nat -F
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -t nat -A PREROUTING -p tcp --dport 445 -j DNAT --to-destination :1445
iptables -t nat -A PREROUTING -p tcp --dport 139 -j DNAT --to-destination :1139
iptables -t nat -A PREROUTING -p tcp --dport 137:139 -j DNAT --to-destination :1137-1139
iptables -t nat -A PREROUTING -p udp --dport 137:139 -j DNAT --to-destination  :1137-1139
```

Sometimes for testing purpose, it is required to access CIFS from the local server machine. In this case, the above configuration is insufficient and it is required to execute the following commands:

```
iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 445 -j REDIRECT --to-port 1445
iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 139 -j REDIRECT --to-port 1139
iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 137 -j REDIRECT --to-port 1137
iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 138 -j REDIRECT --to-port 1138
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OpenKM Configuration

To configure it go to ***Administration > Confguration*** *and change the following properties:*

Field / Property	*Type*	*Description*	*Value*
system.cifs.server	***Boolean***	*Property to enable or disable CIFS.*	*true*
system.cifs.authenticator.class	***String***	*Class used by CIFS to authenticate against OpenKM.*	*com.openkm.integration.cifs.authenticator.DatabaseCIFSAut* *Depending the on authentication adapter values should be us* *authenticar class:* *com.openkm.integration.cifs.authenticator.Datab* *com.openkm.integration.cifs.authenticator.LdapC*
system.cifs.debug	***Boolean***	*Property used to allow detailed log of CIFS functionality.*	*false*
system.cifs.domain.name	***String***	*Domain or workgroup where OpenKM server is installed.*	
system.cifs.hostname	***String***	*Hostname where OpenKM is installed.*	*0.0.0.0*
system.cifs.port.datagram	***Integer***	*Port number used by datagrams.*	*By default* ***1138***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

system.cifs.port.name	***Integer***	*Port number used by CIFS.*	*By default* ***1137***
system.cifs.port.session	***Integer***	*Port number used by session.*	*By default* ***1139***
system.cifs.port.smb	***Integer***	*Port number used by SMB.*	*By default* ***1445***
system.cifs.shared.name	***String***	*Property to inidicate the name of the repository*	*By default the name is* ***repository***

After changing these properties OpenKM must be restarted.

If it is not a fresh installation of OpenKM, it is important to reset user passwords from OpenKM administration Configuration (no mather if it is the same password) in order to generate ***md4 version*** *needed for authentication.*

Windows client

It is necessary to modify a ***Local Security Policy****. To do it, go to* ***Controls Panel*** *→* ***Administrative Tools*** *→* ***Local Security Policy*** *→* ***Local Policies*** *→* ***Security Options****:*

Option Network secutiry: LAN Manager authentication Level and select: ***Send LM & NTLM: Use NTLMv2 session cecurity if negotiated.***

You can also do it by lauching the following command from command line:

REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "LmCompatibilityLevel" /t "REG_DWORD" /d "1" /f

Then to connect to the remote machine, go to Computer → Map Network Drive:

- *Drive: Any letter available*
- *Folder: \\server_ip\repository OR \\server_name\repository*
- *Connect using different credentials (chek it if the Windows user and password are different to OpenKM user).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Then click on finish button.

If connect using different credentials was checked, a new window where introduce the corresponding credentials will appear.

You can also do it by lauching the following command from command line:

net use <driver letter> *****<servername>********<sharename>* ***/user:<username>*** ***<password>*** *and as an example:* ***net use Z: \\server_ip\repository OR \\server_name\repository /user:okmAdmin admin_password***

Active Directory Authentication

In this case, it is considered that the system was integrated previously with Active directory. In this case it will be necessary to configure:

- *Active Directory*
- *OpenKM Server*
- *OpenKM configuration*
- *Windows client*

Active Directory

It is necessary to create an user that OpenKM, using a private key, will use to validate the credentials of the incoming CIFS request. To do it, the next steps must be followed:

1. *Create a Computer Object that will be the one corresponding to the OpenKM server.*
2. *Once created, modify its properties by enabling the property Trusted for delegation.*
    1. *Trusted for delegation. This step is important because, if not enabled, it will not be possible negotiate among client and server.*
3. *The server must authenticate against Active Directory before validating the tickets sent by the client.This authentication is performed using a file (keytab) that map the service (application server) with a domain user. This user must be used only and exclusively for this purpose because if this user is used to logon in another computer, this user will not be able to be used to logon the application server. Once created the user, its properties are modified to not allow the modification of the password and the non expiration of the account. Finally, select the Account tab and enable* ***Do not required Kerberos preauthentication option*** *in the Account Options selection. For this example, it is created a user call* ***servercifs****.*

a. *Select the Account tab and enable the Do not require Kerberos preauthentication option in the Account Options section.*

1. *Generate keytab file. This file is an encoded copy of the private key of an user in kerberos. It is used by the web application as an authentication proxy of the credentials sent by the user. The following command is executed:*

```
ktpass -princ cifs/tomcat.example.com@EXAMPLE.COM -mapuser servercifs@EXAMAPLE.
       -ptype KRB5_NT_PRINCIPAL -pass * -out cifs.keytab -kvno 0
```

- *it is important to highlight :*
  1. *tomcat.example.com@EXAMPLE.COM: full name of the tomcat server and its domain.*
  2. *servercifs@EXAMAPLE.COM : user created previously.*
  3. *cifs.keytab: file where it is exported the private key that will be used by the Tomcat server.*

2. *A Service Principal Name (SPN) needs to be setup with CIFS and a server name: in example: tomcat.example.org where tomcat servlet container is running. This is used with domain user and its keytab is then used as a service credential:*

```
C:\> setspn -A cifs/tomcat.example.org servercifs

C:\> setspn -A cifs/tomcat.example.org servercifs@EXAMPLE.COM

# where tomcat.example.org is the machine where tomcat servlet container is runr
# servercifs is the user previously created.
```

> *Where tomcat.example.org is the machine where tomcat servlet container is running and user is the user previously created.*
>
> *Usually the previous command "ktpass" should create the service cifs/tomcat.example.org, when it's created will be raised an error indicating is yet present.*

OpenKM server in Active Directory configuration

The following steps must be folowed:

Introduce server in domain

The first thing to do is to include the application server inside the domain, if it was not done previously. To do it follow the guide:

http://cerowarnings.blogspot.com.es/2011/11/how-to-linux-en-dominio-windows.html

One thing to take into account is to add the following configuration to the file /etc/host due to previously known problems:

```
# ldap in /etc/hosts
192.168.1.110 example.com
192.168.1.110 Schema.Configuration.example.com
192.168.1.110 Configuration.example.com
192.168.1.110 DomainDnsZones.example.com
192.168.1.110 ForestDnsZones.example.com
```

Redirect the required ports

If the system could not be started with root permissions the ports commented bellow, must be over 1000 (the default system

ports are 138,137,139,445), so it is necessary to redirect the original ports to the new ones. To do this it is necessary to execute the following commands as root or using sudo:

```
$ modprobe iptable_nat
$ iptables -F
$ iptables -t nat -F
$ iptables -P INPUT ACCEPT
$ iptables -P FORWARD ACCEPT
$ iptables -P OUTPUT ACCEPT
$ iptables -t nat -A PREROUTING -p tcp --dport 445 -j DNAT --to-destination :1445
$ iptables -t nat -A PREROUTING -p tcp --dport 139 -j DNAT --to-destination :1139
$ iptables -t nat -A PREROUTING -p tcp --dport 137:139 -j DNAT --to-destination :1137-1139
$ iptables -t nat -A PREROUTING -p udp --dport 137:139 -j DNAT --to-destination  :1137-1139
```

Sometimes for testing purpose, it is required to access CIFS from the local server machine. In this case, the above configuration is insufficient and it is required to execute the following commands:

```
$ iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 445 -j REDIRECT --to-port 1445
$ iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 139 -j REDIRECT --to-port 1139
$ iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 137 -j REDIRECT --to-port 1137
$ iptables -t nat -A OUTPUT -p tcp -d 127.0.0.1 --dport 138 -j REDIRECT --to-port 1138
```

kerberos configuration file

Install kerberos client:

```
$ apt-get krb5-user krb5-config
```

Before starting the application Kerberos must be configured properly to validate tickets correctly. In Linux systems this file is located at "/etc/kbr5.conf".

In this file Domain, KDC server and allowed algorithms must be specified. The most importans algorithms are ***arcfour-hmac-md5*** *and* ***rc4-hmac*** *which Microsoft Active Directory offers.*

Preservate almost krb5.con default configuration and make only the suggested changed shown below.

```
[libdefaults]
default_realm = EXAMPLE.COM
#these are mandatory for kerberos.
permitted_enctypes = aes128-cts rc4-hmac
default_tgs_enctypes = aes128-cts rc4-hmac
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
default_tkt_enctypes = aes128-cts rc4-hmac
[realms]
EXAMPLE.COM = {
  kdc = SERVER.example.com
  admin_server = SERVER.example.com
 default_domain = EXAMPLE.COM
}
[domain_realm]
 example.com = EXAMPLE.COM
 .example.com = EXAMPLE.COM
```

cifs.login.config File

Create a configuration file needed to enable java login with content:

```
JLANServerCIFS{
com.sun.security.auth.module.Krb5LoginModule required
storeKey=true
useKeyTab=true
keyTab="/path/to/cifs.keytab"
principal="cifs/tomcat.example.com";
};
```

As can be observed:

- *JLANServerCIFS correspond with the property* ***system.cifs.ldap.login.entry***
- *Property principal correspind to the property:* ***system.cifs.ldap.principal***
- *Property keytab it is the secure place where the previously created* ***.keytab file*** *that contains the key to access Active directory has been copied.*

> *The file is usually found in the folder* ***JRE/lib/security.*** *However it could be placed in any place with a minimum security.*

Enable Java login configuration.

To enable java login, it is necessary to change the configuration of the Java security file that it is usually place at:

JRE/lib/security/java.secutity and add the following line:

```
login.config.url.1= file:${java.home}/lib/security/cifs.login.config where cifs.login.config is the file created in the prevous section.
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*The default value **${java.home}/lib/security** could be change to any other route where the previous created file jlan.login.config has been stored.*

OpenKM Configuration in Active Directory Configuration

*To configure it : go to **Aministration** > **Confguration** and change the following properties :*

Field / Property	*Type*	*Description*	*Value*
system.cifs.server	***Boolean***	*Property to active or deactive CIFS*	*true*
system.cifs.authenticator.class	***String***	*Class used by CIFS to authenticate against OpenKM*	*com.openkm.integration.cifs.authenticator.LdapCIFSAuthe* *Depending the on authentication adapter values should be authenticar class:* *com.openkm.integration.cifs.authenticator.Dat* *com.openkm.integration.cifs.authenticator.Lda*
system.cifs.debug	***Boolean***	*Property used to allow detailed log of CIFS functionality*	*false*
system.cifs.domain.name	***String***	*Domain or work group where OpenKM server is installed*	*EXAMPLE*
system.cifs.hostname	***String***	*Hostname where OpenKM is installed*	*0.0.0.0*
system.cifs.ldap.kdc	***String***	*Name of the Active*	*server.example.com*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		directory server	
system.cifs.ldap.principal	***String***	*Service created previously in Active Directory*	*cifs/tomcat.example.com*
system.cifs.ldap.password	***String***	*Password of the user created for authentication purpose*	*In the example the user was servercifs*
system.cifs.ldap.realm	***String***	*Name of the domain where Active directory is installed*	*EXAMPLE.COM*
system.cifs.ldap.login.entry	***String***	*Kerberos configuration name inside configurations file*	*JLANServerCIFS*
system.cifs.port.datagram	***Integer***	*Port number used by datagrams*	*By default **1138***
system.cifs.port.name	***Integer***	*Port number used by cifs*	*By default **1137***
system.cifs.port.session	***Integer***	*Port number used by session*	*By default **1139***
system.cifs.port.smb	***Integer***	*Port number used by smb*	*By default **1445***
system.cifs.shared.name	***String***	*Property to inidicate the*	*By default the name is **repository***

		name of the repository	

After changing this properties, OpenKM must be restarted.

Windows client in Active Directory configuration

It is necessary to modify a ***Local Security Policy****. To do it, go to* ***Controls Panel*** *→* ***Administrative Tools*** *→* ***Local Security Policy*** *→* ***Local Policies*** *→* ***Security Options****:*

Option Network secutiry: LAN Manager authentication Level and select: ***Send LM & NTLM: Use NTLMv2 session cecurity if negotiated.***

You can also do it by lauching the following command from command line:

REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "LmCompatibilityLevel" /t "REG_DWORD" /d "1" /f

Then to connect to the remote machine, go to Computer → Map Network Drive:

- *Drive: Any letter available*
- *Folder: \\server_name\repository*
- *Connect using different credentials (chek it if the Windows user and passwords are different to OpenKM user).*

Then click on finish button.

If connect using different credentials was checked, a new window where introduce the corresponding credentials will appear.

You can also do it by lauching the following command from command line:

net use <driver letter> \\<servername>\<sharename> /user:<username> <password> *and as an example:* ***net use Z: \\server_name\repository /user:okmAdmin admin_password***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Cache configuration

OpenKM uses several caches to enhance performance. Some of these caches are directly used by OpenKM and others indirectly, for example those used by Hibernate. All these caches have default values which make them useful but these default values are not always good for every installation. The cache engine used is the very popular EHCache, which is widely used and very configurable. You can see the default values at ***$TOMCAT_HOME/webapps/OpenKM/WEB-INF/classes/ehcache.xml****.*

Don't modify this file because all changes will be lost when OpenKM is restarted or updated.

If you want to tweak this default configuration you have to duplicate the provided configuration file into ***$TOMCAT_HOME/ehcache.xml*** *and make the proper changes to match your requirements. This is a sample definition:*

```
<cache name="com.openkm.cache.ldapPrincipalAdapter.general"
       maxElementsInMemory="1000"
       timeToIdleSeconds="300"
       timeToLiveSeconds="0"
       overflowToDisk="true"
       eternal="false" />
```

The most important values are:

- ***timeToLive****: The maximum number of seconds an element can exist in the cache regardless of use. The element expires at this limit and will no longer be returned from the cache. The default value is 0, which means no TTL eviction takes place (infinite lifetime).*
- ***timeToIdle****: The maximum number of seconds an element can exist in the cache without being accessed. The element expires at this limit and will no longer be returned from the cache. The default value is 0, which means no TTI eviction takes place (infinite lifetime).*

Be careful with ***eternal****, because it sets whether elements are eternal. In case of true, timeouts are ignored and the element never expires.*

So, in this case the cache called "com.openkm.cache.ldapPrincipalAdapter.general" will have a maximum of 1000 elements in memory and in case of overflow will be written to disk. These elements will be removed from this cache if are unused for 5 minutes. Otherwise, will remain forever. If you want to force a refresh, you have to change the ***timeToLiveSeconds*** *to 3600 and will be refreshed every hour.*

You can check these caches status by going to ***Administration*** *>* ***Utilities*** *>* ***Cache stats****. In this part you can enable and see the cache usage stats and even reset a give one. More info at Cache stats.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring CADViewer

CADViewer is a third-party product which can be integrated into OpenKM to provide previsualization of AutoCAD documents.

> *The required execubles should be already included in your OpenKM installation. Please, check if the **$TOMCAT_HOME/bin/cadviewer** folder exists and contains executable files like **ax2020** and **dwglist**, among others.*

*Go to **Administration** > **Utilities** > **Database query**. At bottom right select option **JDBC**, paste into the box the SQL insert:*

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('8e642680-0dcd-4ded-a501-2aa3e
```

Some configuration properties are needed:

Field / Property	*Type*	*Description*
extension.cadviewer.autoxchange	***String***	*Path to **autoxchange** utility. Used in thumbnail generation, preview and PDF conversion.*
extension.cadviewer.linklist	***String***	*Path to **linklist** utility. Extract text and metadata from DWG documents.*
extension.cadviewer.dwglist	***String***	*Path to **dwglist** utility. Get XREFs from DWG documents. Also used in thumbnail, preview and PDF conversion.*
extension.cadviewer.sibling.xrefs	***Boolean***	*When this parameter is set to **true** the AutoCAD files's references have to be all in the same folder. If it is set to **false** the AutoCAD files's references can be stored in any place in the taxomony tree. In this case, keep on mind that you should avoid document name duplication. There is an automation called **CheckUniqueDocumentName** which can help you to maintain this restriction.*
extension.cadviewer.debug	***Boolean***	*Set to **true** to view the commands executed to conversion in the log file. Only for debug purpose.*
extension.cadviewer.remove.temp.files	***Boolean***	*Set to **false** to keep the temporary files creating during*

		conversion. Only for debug purpose.
extension.cadviewer.qr.url	***String***	*The URL of the QR code application:* ***http://openkm.your-domain.com/qrcad/***
extension.cadviewer.qr.user	***String***	*Name of the user of the QR code application.*
extension.cadviewer.qr.password	***Password***	*The password of the user of the QR code application.*
extension.cadviewer.qr.pos_x	***Integer***	*Percent X from the lower-left where to put the QR code in the AutoCAD file.*
extension.cadviewer.qr.pos_y	***Integer***	*Percent Y from the lower-left where to put the QR code in the AutoCAD file.*
extension.cadviewer.qr.width	***Integer***	*QR code width percent.*

CADViewer has two kind of licenses:

- ***axlic.key**: This is the license for the executables. This one should be placed in the same directory of all these executables.*
- ***cvlicense.js**: Used in the Javascript CAD previewer. Should replace the default license at **$TOMCAT_HOME/webapps/openkm/preview/cadviewer/js** folder.*

*In addition, you have to configure a patch to preserve this license across future OpenKM updates. To do this, create a folder called **$TOMCAT_HOME/patch** and place the license at **$TOMCAT_HOME/patch/preview/cadviewer/js**. Once copied, create an script called **prepare.sh** with this content:*

```
#!/bin/bash

if [ -f "openkm.war" ]; then
  echo "Patching WAR..."
  jar uf openkm.war preview/cadviewer/js/cvlicense.js
  echo "Done!"
else
  echo "Not able to patch file"
fi
```

Remember to check executable permissions of the prepare.sh script and the CADViewer's executables.

If you have problems with the conversion, try to execute from the command line. For example, this would make a PDF conversion:

```
$ autoxchange -i=test.dwg -f=pdf
```

In case of Linux, ensure you have configured the LD_LIBRARY_PATH properly. In case of Windows the program may fail because VCRUNTIME140.DLL is missing. If you see this error, download the Microsoft Visual C++ 2015 Redistributable from https://www.microsoft.com/en-us/download/details.aspx?id=52685 and install it.

To view external references for each AutoCAD document must set the value of the configuration key ***extension.external.tab.document****. For do it, go to* ***Administration > Configuration*** *and set the appropiate value for this key:*

```
extension.external.tab.document = References;/openkm/frontend/XRefs
```

You can use this SQL sentence to set the configuration property. Go to ***Administration*** *>* ***Utilities*** *>* ***Database query*** *and execute:*

```
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('extension.external.tab
```

To enable this extension go to ***Administration > Utilities > Database query****. At bottom right select option* ***JDBC****, paste into the box the SQL insert:*

```
INSERT INTO OKM_EXTENSION (EXT_UUID, EXT_NAME) VALUES ('38a9d032-30b9-470d-93ad-47f5e
```

Configuring Automations:

Name	*Event*	*At*	*Validations*	*Actions*
CAD Create	***Document create***	***post***	***Type:*** *NameMatches* ***Param 0:*** *.+\.[dD][wW][gG]*	***Type:*** *NewCadReferences* ***Order:*** *0*
CAD Update Pre	***Document update***	***pre***	***Type:*** *NameMatches* ***Param 0:*** *.+\.[dD][wW][gG]*	***Type:*** *DelCadReferences* ***Order:*** *0*
CAD Update Post	***Document update***	***post***	***Type:*** *NameMatches* ***Param 0:*** *.+\.[dD][wW][gG]*	***Type:*** *NewCadReferences* ***Order:*** *0*
CAD Delete	***Document delete***	***pre***	***Type:*** *NameMatches* ***Param 0:*** *.+\.[dD][wW][gG]*	***Type:*** *DelCadReferences* ***Order:*** *0*
CAD Folder Delete	***Folder delete***	***pre***	***Type:*** *PathContains* ***Param 0****: /okm:root*	***Type****: DelCadReferences* ***Order****: 0*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring VidSigner

VidSigner is a third-party product which can be integrated into OpenKM to allow digital signature of a document.

Some configuration properties are needed:

Field / Property	*Type*	*Description*
vidsigner.user	***String***	*Name of the user given for the subscription to VidSigner.*
vidsigner.password	***Password***	*Password of the user.*
vidsigner.device.name	***String***	*Name of the device given by VidSigner where we want to send the document to be signed. Don't set if want to sign the document in any device associated with the subscription. In this case, you can find the document in the device, enabling search mode in configuration and finding by OpenKM user.*
vidsigner.notification.url	***String***	*The URL where the notification will be sent:* ***http://openkm.your-domain.com/openkm/services/rest/signed***
vidsigner.preproduction.mode	***Boolean***	*Set to true to test vidsigner in pre production mode.*
vidsigner.sign.page	***String***	*The page where to show the signature image. If it is set to zero then the signature will be invisible.*
vidsigner.sign.posX	***String***	*The X position in millimeters where to place the signature image.*
vidsigner.sign.posY	***String***	*The Y position in millimeters where to place the signature image.*
vidsigner.sign.sizeX	***String***	*The X size in millimeters of the signature image.*
vidsigner.sign.sizeY	***String***	*The Y size in millimeters of the signature image.*
vidsigner.generate.log	***Boolean***	*Set to true to generate a detail log with all the information of the call to VidSigner service*

In addition, you have to create a user in OpenKM with the name and password given by VidSigner to your subscription.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Also, you have to create two new Automations on event ***Set metadata group*** *(at post) and associate to the first one the action* ***VidSigner Document To Sign*** *and to the second the action* ***VidSigner Document Signed***.

Finally, you have to register this group of metadata:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.8//EN"
                                 "http://www.openkm.com/dtd/property-groups-2.8.dtd">
<property-groups>
<property-group label="ViDSigner" name="okg:vidsigner">
   <checkbox label="Pending to sign" name="okp:vidsigner.pending" value="true" />
   <input label="Document GUI" name="okp:vidsigner.docgui" readonly="true" />
</property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring search highlighter

The search highlighter feature shown a highlight text in the search view.

By default search highlighter is disabled. To enable you must add in the OpenKM.cfg the parameter ***hibernate.search.highlighter***.

```
hibernate.search.highlighter=true
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring email digital signature

A digital signature attached to an email message increases security by providing information to the receiver that you are the right person and no other person who signed the contents of the email message. Your digital signature, which includes your certificate and public key is part of your digital ID. This digital ID identifies your unique digital mark and signals the recipient that the content hasn't been altered in transit.

Secure/Multipurpose Internet Mail Extensions (S/MIME) is a method of exchanging emails securely. With the S/MIME technology, a digital signature can be appended to email messages to ensure non-repudiation and data integrity. OpenKM since v7.1.14 is able to sign email messages using S/MIME.

These are the involved configuration properties:

Field / Property	*Type*	*Description*
mail.smime.certificate.file	***String***	*The location of the S/MIME certificate.*
mail.smime.certificate.password	***String***	*The password to access the S/MIME certificate.*

Once these configuration properties are set, the emails sent from OpenKM will be signed.

More info

- *Getting an SMIME certificate*
- *Wikipedia S/MIME*

Self-signed certificates are not recommended because won't be automatically validated at destination. You can find more information at:

- *Can I create my own S/MIME certificate for email encryption?*
- *Create Self-Signed S/MIME Certificates*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring anonymous user

Behind the anonymous user, there's a standard user named "public" which autologin to the application using a special URL.

Follow these steps to configure:

- ***Create a user** named "**public**".*
- ***Assign** at least the role named "**ROLE_USER**" to this user.*
- *Login with the user "**public**" in the standard UI to verify which nodes and options have access.*

The "public" user must have READ access to the main root nodes (okm:root, okm:templates, etc.).

*A **good practice** is to **create a role** named "**ROLE_PUBLIC**" and **assign read access to the public nodes (** documents, folders, etc.). Remember to **assign the role** to the user "**public**".*

*Another option is to **assign the user** "**public**" to these nodes.*

Optional steps:

- *Create a profile for the "public" user. Disable all the features you consider should not have access.*

Finally, share the autologin public URL which will be something like [*http://yourhost:8080/openkm/public/*](http://yourhost:8080/openkm/public/)

In case you want to logout the public user, please use a URL like http://yourhost:8080/openkm/kcenter/#/logout

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Throubleshooting

Here you can find some common problems and how to solve them. The context is grouped into several sections:

- *Server configuration*
- *Client configuration*
- *Browser configuration*

In special, take a look to ***Java and Chrome problem*** *section into Browser configuration which will help you to get Java applets working in Google Chrome.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Server configuration

Packet for query is too large (MySQL)

If you get an error like this:

> *Packet for query is too large (1708726 > 1048576).*
>
> *You can change this value on the server by setting the 'max_allowed_packet' variable.*

You need to modify your MySQL server configuration file and increase the value of the ***max_allowed_packet*** *property. Don't forget to restart the MySQL after any change in this file. In Debian based distros like Ubuntu, this configuration file is located at /etc/mysql/my.cnf.*

Not able to connect from other computer in the network

Ensure you have access from the other computers to the server.

Ensure your server firewall allows remote connection to the destination port.

*If you expose the tomcat port 8080, ensure is binding your ip, the "**address**" parameter value should be "**0.0.0.0**" at $TOMCAT_HOME/conf/server.xml:*

```
<Connector port="8080" address="0.0.0.0" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

After any modification in server.xml to take effect must restart the tomcat.

Error downloading document as PDF

This error usually is caused by OpenOffice or LibreOffice service configuration.

*Check if the OpenOffice or LIbreOffice service is started by OpenKM. You should get a program named "**soffice.bin**" running after application finished the startup.*

IMAP SSL certificate error

First obtain the certificate with the openssl command and after this import the certificate with:

```
$ keytool -import -alias mail.yourcompany.com -keystore /etc/java-6-sun/security/cacerts -file imapd.pem
```

The application must be restarted to refresh the SSL certificates. More information at Connecting+to+SSL+services.

OpenKM server has no direct Internet connection

*If you need a proxy to access the Internet, edit your "**/etc/bash.bashrc**" file as root and put these lines at the end of your "**/etc/bash.bashrc**" file:*

```
export http_proxy=http://username:password@proxyserver.net:port/
export ftp_proxy=http://username:password@proxyserver.net:port/
```

You can omit the username:password, if your proxy server has no password.

Access denied installing OpenKM as service on Windows 7

*The command shell works slightly differently in Vista. By default when you run cmd.exe, it does not give administrator rights. To get around it, **create a shortcut to cmd.exe on desktop, open Properties > Compatibility tab and check box for Administrator rights**. Then you can do admin stuff on cmd line.*

Problems with character encodings

If your language has non-ASCII characters and they are displayed as garbage you have a problem with encodings.

*Check the $TOMCAT/bin/setenv.sh or $TOMCAT/bin/setenv.bat file. You should have set the JVM parameter "**-Dfile.encoding=utf-8**".*

> *The application must be restarted.*

PermGen errors

> *Since Java 8 there is no such **PermSize** and **MaxPermSize** options.*

If you see this error:

```
java.lang.OutOfMemoryError: PermGen space
```

*Edit your "**$TOMCAT_HOME/bin/setenv.sh**" (or "**$TOMCAT_HOME/bin/setenv.bat**" if using Windows) and increase the value of the JVM parameter "-XX:MaxPermSize":*

```
JAVA_OPTS="-Xms256m -Xmx2048m -XX:PermSize=128m -XX:MaxPermSize=512m"
```

See also Presenting the Permanent Generation.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

LDAP Sizelimit exceeded

Caused because bindDN has size limit for each ldap query.

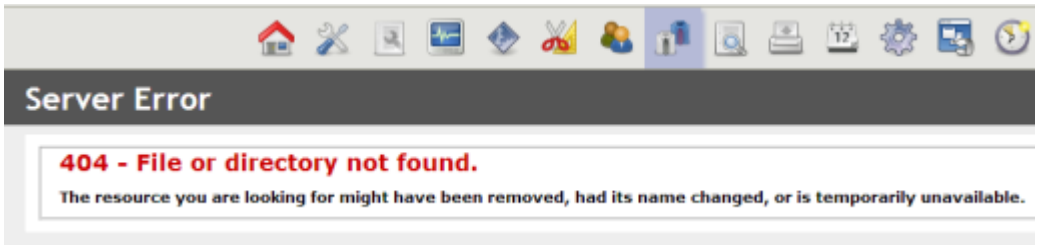
In Openldap The solution is to increase the limit of bindDN, for example (to change size limit to user used by openkm ldap connection)

```
limits dn.exact="cn=reader,ou=accounts,ou=admins,dc=example,dc=com" size=100000
```

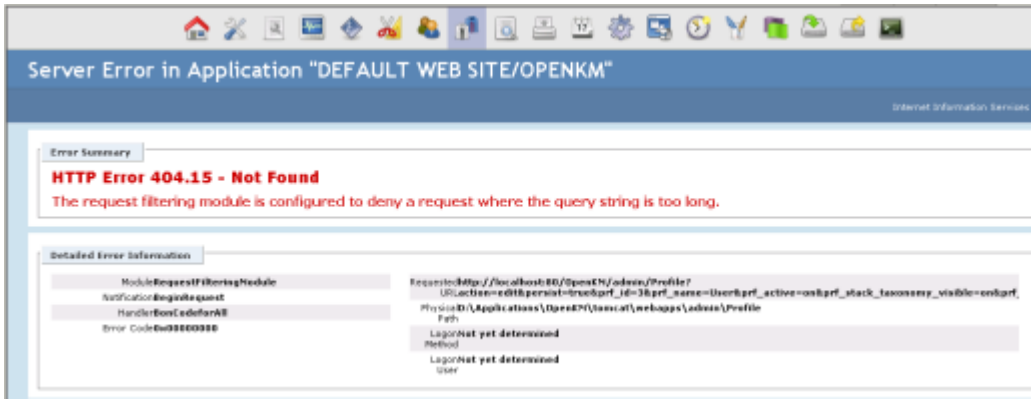
See also OpenLDAP Limits.

Behind IIS proxy you get error 404 or 404.15

Using IIS as a web server by default it has a limit for the query string sent to the server if limit is passed then you get some error screen on the client side:

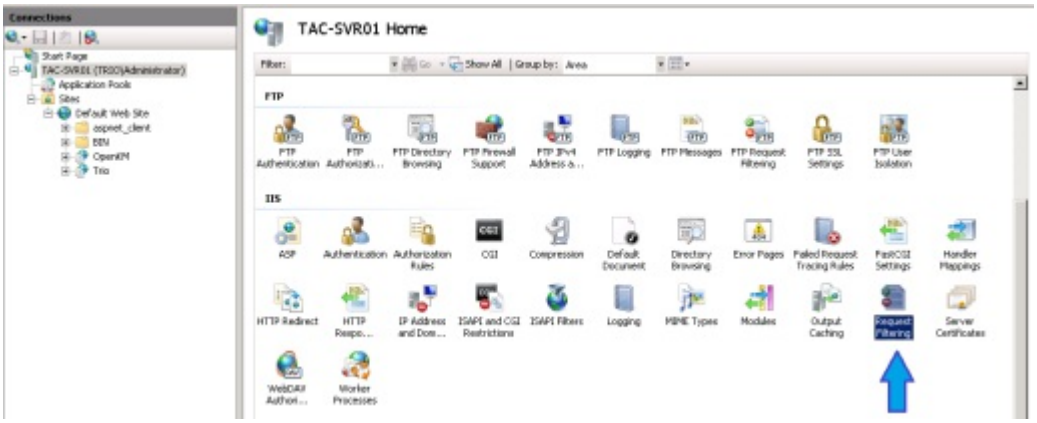


In the server side we get a more detailed error:



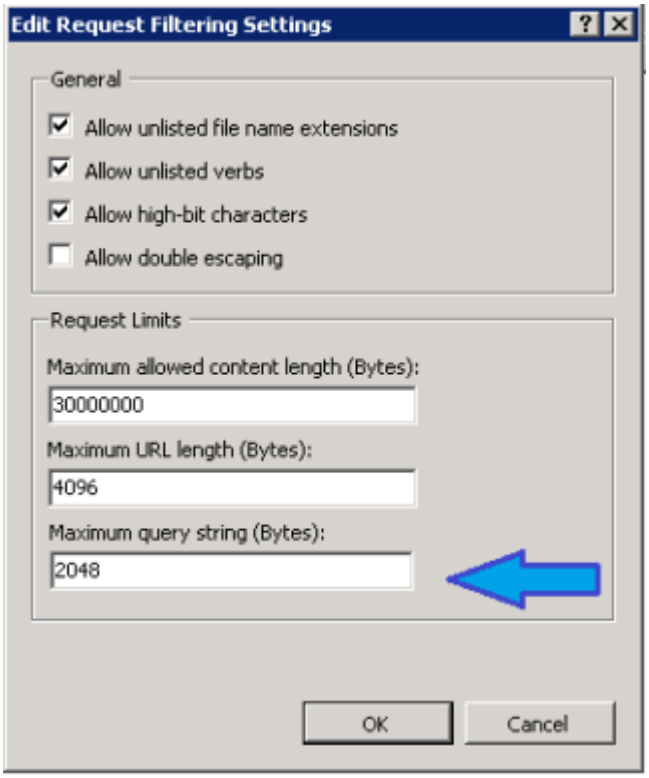
To solve this issue we need to increase the maximum size of the query string on the web server:

- ***Open Internet Information Services (IIS) Manager***.
- *In the **Connections pane**, go to the **connection > site > application**, or directory for which you want to modify your request filtering settings.*
- *In the **Home pane, double-click Request Filtering**.*



*Click **Edit Feature Settings...** in the Actions pane.*

Change the maximum query string length:

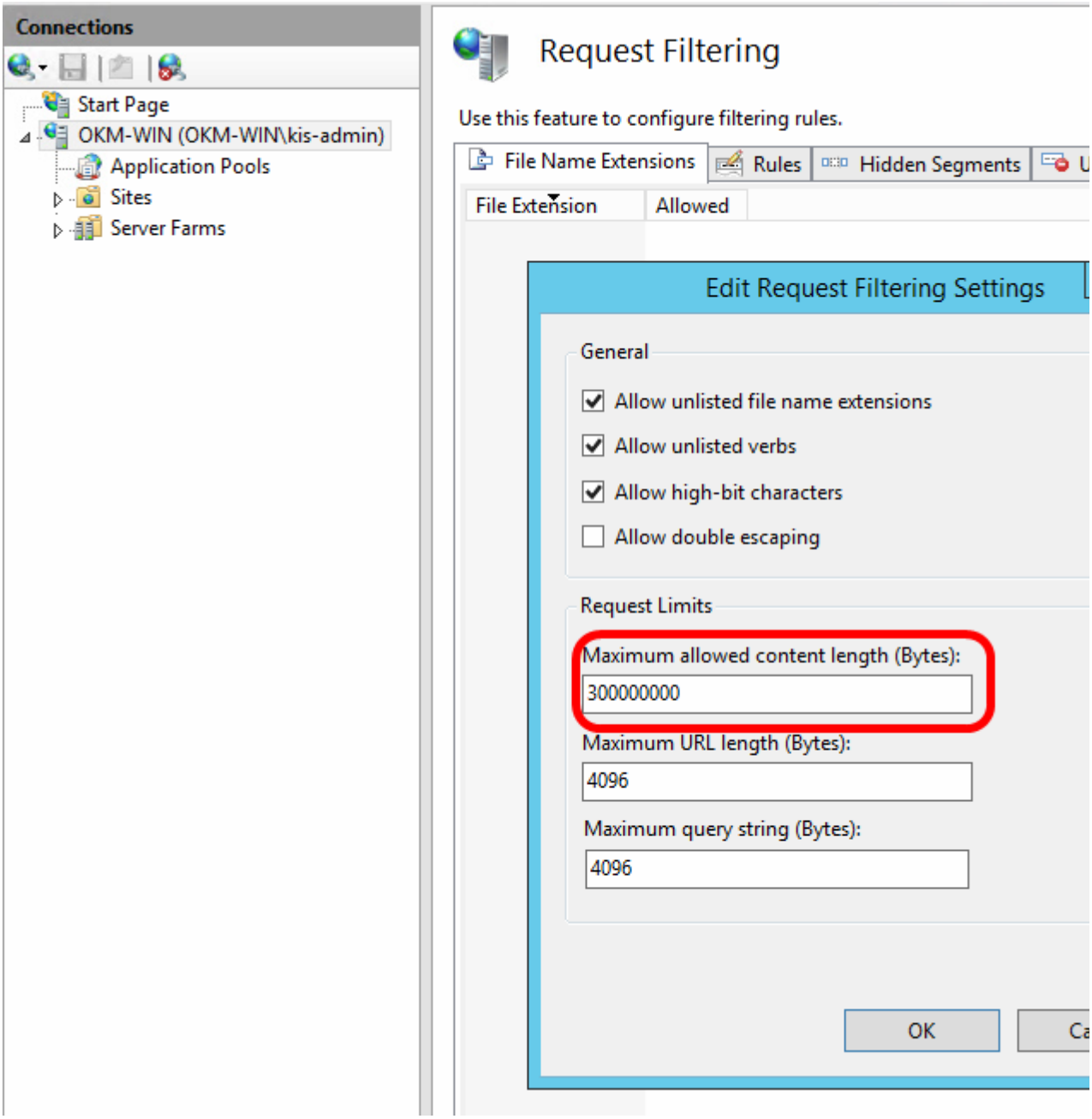


More information at [OpenKM forum](#).

IIS gets IOException: Internal Server Error while uploading file

Solution one

Change the "Maximum allowed content lenght (Bytes)" from your IIS configuration parameters:



Solution two

- ***Go to webapps*** *folder*
- ***Edit*** *the* ***web.config*** *file and* ***add the following under <configuration> where "2097151"*** *is the maximum allowed in IIS:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<system.web>
   <httpRuntime maxRequestLength="2097151" />
</system.web>
```

If there is no web.config then just create one from text file and add the following :

```
<configuration>
   <system.web>
      <httpRuntime maxRequestLength="2097151" />
   </system.web>
</configuration>
```

Zoho fails with SSL connection

To use Zoho integration you have to configure your server to deliver the whole certificate chain by itself, because the Zoho client does not looks for the chain. Our solution is reconfigure APR in Tomcat to deliver the CA chain separatly to the server certificate.

*The configuration file at "**$TOMCAT_HOME/conf/server.xml**" it looks like this:*

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443
     This connector uses the JSSE configuration, when using APR, the
     connector should be using the OpenSSL style configuration
     described in the APR documentation -->

   <Connector port="8443" address="0.0.0.0" protocol="HTTP/1.1" SSLEnabled="true"
              maxThreads="150" scheme="https" secure="true" acceptCount="100"
              enableLookups="false" clientAuth="true" sslProtocol="SSLv3"
              SSLCipherSuite="ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:!LOW:!SSLv2:!EXPORT"
              SSLCertificateFile="/home/openkm/tomcat-7.0.27/cert/serverpublic.pem"
              SSLCertificateKeyFile="/home/openkm/tomcat-7.0.27/cert/serverprivate.pe
              SSLCertificateChainFile="/home/openkm/tomcat-7.0.27/cert/serverpublicch
              allowUnsafeLegacyRenegotiation="false"
          />
```

More information at our Forum.

CMIS bad URL behind Nginx proxy

Use this configuration:

```
location /OpenKM/ {
   proxy_pass http://127.0.0.1:8080/OpenKM/;
   proxy_redirect off;

   proxy_set_header Host $host;
   proxy_set_header X-Real-IP $remote_addr;
   proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
```

More information at Forum.

Sigar java.lang.UnsatisfiedLinkError Known issue in Windows

*If you have OpenOffice or LibreOffice configured and you get the error "**sigar java.lang.UnsatisfiedLinkError**"probably you have missing the "**sigar path**" in "**PATH**" environment variable.*

```
Caused by: java.lang.UnsatisfiedLinkError: org.hyperic.sigar.Sigar.getSigNum(Ljava/la
 at org.hyperic.sigar.Sigar.getSigNum(Native Method)
 at org.hyperic.sigar.Sigar.kill(Sigar.java:275)
```

*Edit your general environment variable "**PATH**" and add your "**$TOMCAT_HOME\lib\sigar**" folder, for example like **PATH=...;C:\tomcat-7.0.27\lib\sigar***

The application must be restarted.

WARN net.sf.ehcache.hibernate.strategy.AbstractReadWriteEhcacheAccessStrategy

If you see this message:

```
A soft-locked cache entry was expired by the underlying Ehcache.
If this happens regularly you should consider increasing the cache timeouts and/or ca
```

Do you experiencing any performance or usability problems as a result of this warning? If not then it seems like you should treat this warning as just that: a warning.

*Otherwise, it looks like the solution is to increase the size of **maxElementsInMemory** to account for the difference in what is cached versus what is realistically being used.*

OpenOffice is not starting on Windows

Sometimes is not easy to get OpenKM with soffice service started for several and strange reason which depends on your OS Windows version and other specific configuration done in server.

*First of all ensure your OpenOffice path is correctly configured at **Administration > Configuration parameters >** "**system.openoffice.path**" parameter.*

Ensure there's no other soffice.bin instance running. If there're another soffice.bin instance on execution openkm will not be able to start their own soffice service. Take a look at the process running on Widnows or use a tool like Process Explorer.

Stop firewall and other possible tools which be able to do not allow start soffice service. For example the antivirus.

*Try starting application from command line at "**%TOMCAT_HOME/bin**":*

```
$ catalina.bat run
```

Another option is open a command line with administrator grants. In some Windows OS appears a checkbox to execute with some restrictions, we suggest try both options.

We've observated in some cases the execution with some restrictions goes right and not the case without any restriction.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

RedHat and CentOS issues

CentOS 8: OpenKM not starting as a service automatically

*In CentOS 8, OpenKM service does not start automatically. This is because the security system (**SELinux**) blocks services that are not recognized as native.*

To enable it we must perform the following steps:

- ***Create the openkm.service file within the path (/usr/lib/systemd/system/openkm.service)***

```
$ nano /usr/lib/systemd/system/openkm.service
[Unit]
Description=OpenKM daemon
[Service]
Type=simple
ExecStart=/home/openkm/tomcat-8.5.34/bin/startup.sh
RemainAfterExit=yes
[Install]
WantedBy=multi-user.target
```

***** *ExecStart contains an sample value, the correct path is where the* ***startup.sh*** *file location.*

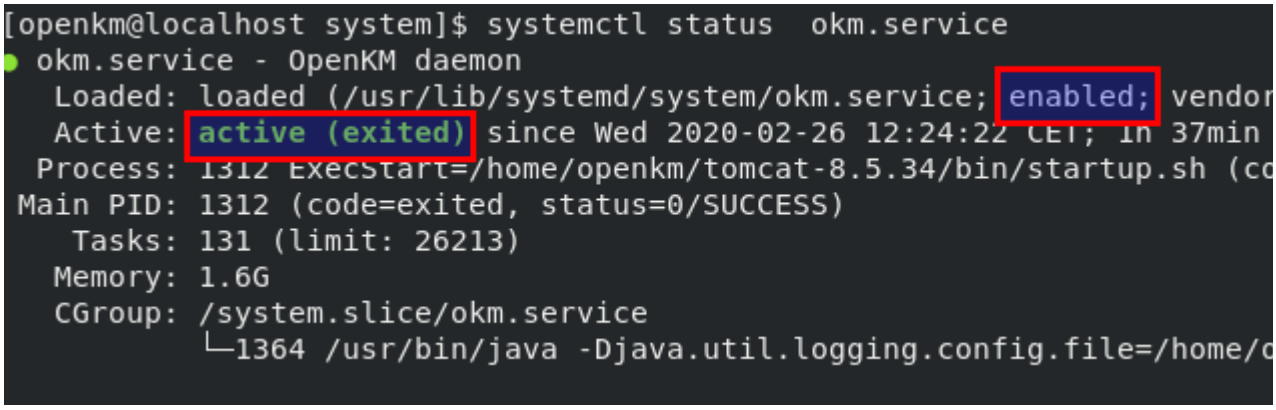
- ***Refresh, enable the service and activate the service.***

```
$ systemctl daemon-reload
$ systemctl enable openkm.service
$ systemctl start openkm.service
```

With the following command we can know the status of the service:

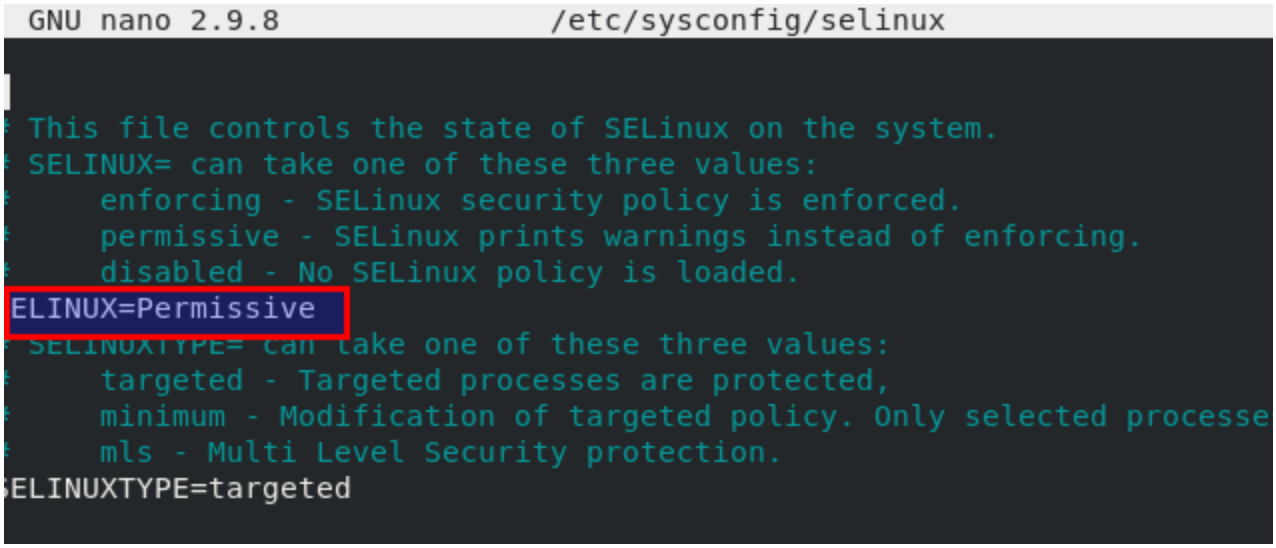
```
$ systemctl status openkm.service
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



- ***Modify the SElinux configuration file:***

```
$ nano /etc/sysconfig/selinux
```



- ***Finally restart the server***

```
$ reboot
```

ImageMagick in CentOS 8

This new CentOS version does not comes with ImageMagick package in the repositories, but we can install a fork called GraphicsMagick from EPEL repository:

```
$ dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm -y
$ dnf install GraphicsMagick
```

Keep on mind that there is no "convert" utility but a general "gm" executable which can be used this way:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
$ gm convert <params>
```

More info at GraphicsMagick convert documentation.

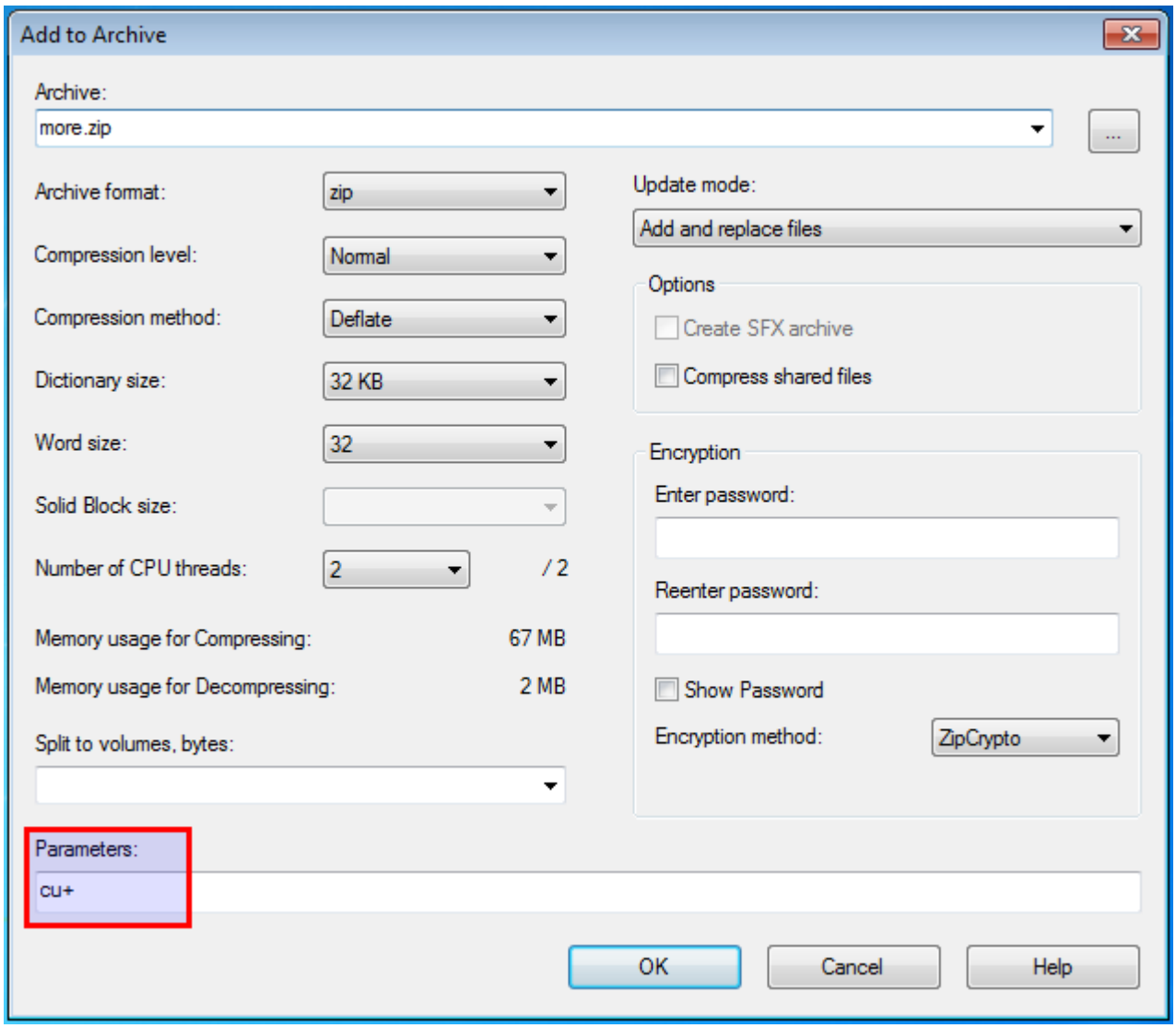
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Client configuration

ZIP filenames with wrong charset

When you download an OpenKM as ZIP maybe the included files have a wrong charset. This only happens with non-latin languages like Greek, Chinese or Arabic. This issue is corrected since OpenKM 4.1 and exported ZIP archives should be opened without problems from WinZip 14 or higher. But UNICODE support in ZIP is a little tricky and may cause problems. To ensure you won't have problems, try using 7-Zip which seems to handle these UNICODE filenames.

The other problem you may have is when you want to import a ZIP into OpenKM. In this case you can also get wrong filenames. Filenames should be encoded in UNICODE, and to ensure it you can force 7-Zip to use this encoding when creating a ZIP:



If you are on Windows you can download 7-Zip from http://www.7-zip.org/. You can also install the port of the 7-Zip compression tool for Linux:

```
$ sudo aptitude install p7zip-full p7zip
```

Microsoft Office Addin disappears after Office restart

Could be a problem with default template (normal.dot). To solve try it:

- *Open normal.dot*
- *Go to menu tools / personalize / Tool bars*
- *Open OpenKM toolbar*
- *Save normal.dot*

Error in Uploader / Scanner applet

To provide more complete information in order to determine the issue, please activate the Java Console. Read How do I enable and view the Java Console? for more info.

Browser configuration

On latest browsers versions the steps described below could not be exactly the same. Take it as a reference of what you should be looking for.

Firefox cause error when click on "Copy to clipboard"

*To enable JavaScript "Copy to clipboard" buttons on Firefox and Mozilla, enter the following line into browser address bar: "**about:config**".*

*The list of preferences should open in browser window. **Right-click somewhere** in the active part of preference list window and choose **New > Boolean** option from context menu. Paste the following Preference name into the New boolean value window input box: "**signed.applets.codebase_principal_support**".*

Unable to run Scanner applet under Linux

If you enable the Java Console and see this error:

```
java.lang.UnsatisfiedLinkError: uk.co.mmscomputing.util.JarLib.loadX: Could not find
```

*You have to install the **libsane-dev** package:*

```
$ sudo aptitude install libsane-dev
```

Internet Explorer error "Document name is different"

*This is a known error in Internet Explorer family. To disable name mismatch check, change the value of configuration parameter named "**system.document.name.mismatch.check**" to false.*

More information at Other configuration parameters

Internet Explorer error downloading files in HTTPS / SSL connections

There's a very old bug from version 5.5 until now that seems like Microsoft has not decided to fix , that causes a problem downloading files with IE behind SSL protocol (the force may be with you, hope some day they will decide to fix it).

*The problem is caused by "**Cache-control: no-cache HTTP header to the response message** ", obviously that's valid for any browser except IE family ... other Microsoft IE hidden feature ?*

Can see some supporting and forum information about it:

- *IE SSL download problem*

- *http://support.microsoft.com/kb/815313*
- *http://support.microsoft.com/kb/323308*
- *http://support.microsoft.com/kb/316431*

The solution is to change a couple of entries in the Windows Registry:

- *Create a file called ss_download_patch.reg*
- *Copy the contents*

```
REGEDIT4
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Internet Settings]
"DisableCachingOfSSLPages"=dword:00000000
"BypassSSLNoCacheCheck"=dword:00000001
"BypassHTTPNoCacheCheck"=dword:00000001
"SSLPragmaNoCache"=dword:00000001
```

- *Execute it (will be imported into Windows Registry)*
- *Then you need to restart Windows for this to take effect.*

Google Chrome and Java applets

As you can read at Java and Google Chrome Browser, Chrome v45 no longer supports NPAPI which is required for Java applets. So you won't be able of using some interesting features in OpenKM like ***Live Edit*** *o* ***Bulk Uploader*** *(both applets). There are a couple of workarounds:*

- *Use another browser like Firefox, which still has NPAPI support.*
- *Go back to Chrome v44 and enable NPAPI support.*

If you prefer keep using Chrome (option 2), follow these steps:

- *Uninstall Google Chrome from your computer (****Control Panel*** *>* ***Uninstall a program****)*
- *Download Google Chrome v44 portable and install it.*
- *Once installed, use this URL to access browser configuration: chrome://flags*
- *Look for* ***Enable NPAPI****, press* ***Enable*** *link and restart Chrome.*
- *Now you would be able to use Java applets again.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Best practices reporting issues

To quickly help you in error identification and resolution, please try to follow the guideline below:

- *Do a **detailed description** of the issue, that will help us understanding the problem do you have.*
- *When possible **make screenshots of all the steps you followed**. The idea is being able to reproduce the error or at least understanding better the context where has occured.*
- *Videos are also welcome.*
- *Share the stack trace error from openkm.log file*

> *Because support website have restriction in document size, share with us the downlaod link with online file sharing tools like:*
>
> - *https://transfer.sh/*
> - *https://filepush.co/*
> - *https://filetransfer.io/*
> - *https://justbeamit.com/*

Getting the stack trace

Linux

*The **openkm.log** file is into the **$TOMCAT/logs folder**. For example, in Linux a common location might be "**/home/openkm/tomcat-8.5.34/logs**".*

To wacth how openkm.log changes in real time you can use the command line:

```
/home/openkm/tomcat-8.5.34/logs:$ tail -f openkm.log
```

To show the last 500 lines of the openkm.log file you can use the command line:

```
/home/openkm/tomcat-8.5.34/logs:$ tail -n 500 openkm.log
```

Windows

*The **openkm.log** file is into the **$TOMCAT/logs folder**. For example, in Linux a common location might be "**c:\tomcat-8.5.34\logs**".*

To wacth how openkm.log changes in real time you can a windows tools like mtail.

> *You have a copy of mtail tools into **c:\tomcat-8.5.34\extras***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Sample of stack trace capture

When you capture the stack trace error it is a good practice include some extra lines before and after the error happens.

```
2019-08-04 12:05:00,020 [Task Scheduler 18] [] INFO  c.openkm.plugin.cron.BaseCronPlu
2019-08-04 12:05:00,032 [Task Scheduler 19] [] INFO  c.openkm.plugin.cron.BaseCronPlu
2019-08-04 12:09:36,471 [http-nio-0.0.0.0-8080-exec-6] [okmAdmin] INFO  com.openkm.ut
2019-08-04 12:09:36,603 [http-nio-0.0.0.0-8080-exec-6] [okmAdmin] ERROR com.openkm.ut
2019-08-04 12:09:36,612 [http-nio-0.0.0.0-8080-exec-6] [okmAdmin] ERROR c.o.s.admin.P
com.openkm.core.ParseException: Element type "hpanel" must be declared.
 at com.openkm.util.FormUtils.parsePropertyGroupsDefinition(FormUtils.java:308)
 at com.openkm.module.db.DbRepositoryModule.registerPropertyGroups(DbRepositoryModule
 at com.openkm.module.db.DbRepositoryModule.registerPropertyGroups(DbRepositoryModule
 at com.openkm.servlet.admin.PropertyGroupsServlet.doPost(PropertyGroupsServlet.java:
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:661)
 at javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at com.openkm.servlet.UserLoggingFilter.doFilter(UserLoggingFilter.java:33)
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at org.springframework.boot.web.filter.ApplicationContextHeaderFilter.doFilterIntern
 at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at org.springframework.boot.actuate.trace.WebRequestTraceFilter.doFilterInternal(Web
 at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter
 at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilte
 at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.j
 at java.lang.Thread.run(Thread.java:748)
2019-08-04 12:10:00,000 [Task Scheduler 18] [] INFO  c.openkm.plugin.cron.BaseCronPlu
2019-08-04 12:10:00,000 [Task Scheduler 6] [] INFO  c.openkm.plugin.cron.BaseCronPlug
2019-08-04 12:10:00,000 [Task Scheduler 25] [] INFO  c.openkm.plugin.cron.BaseCronPlu
2019-08-04 12:10:00,014 [Task Scheduler 6] [] INFO  c.openkm.plugin.cron.BaseCronPlug
```

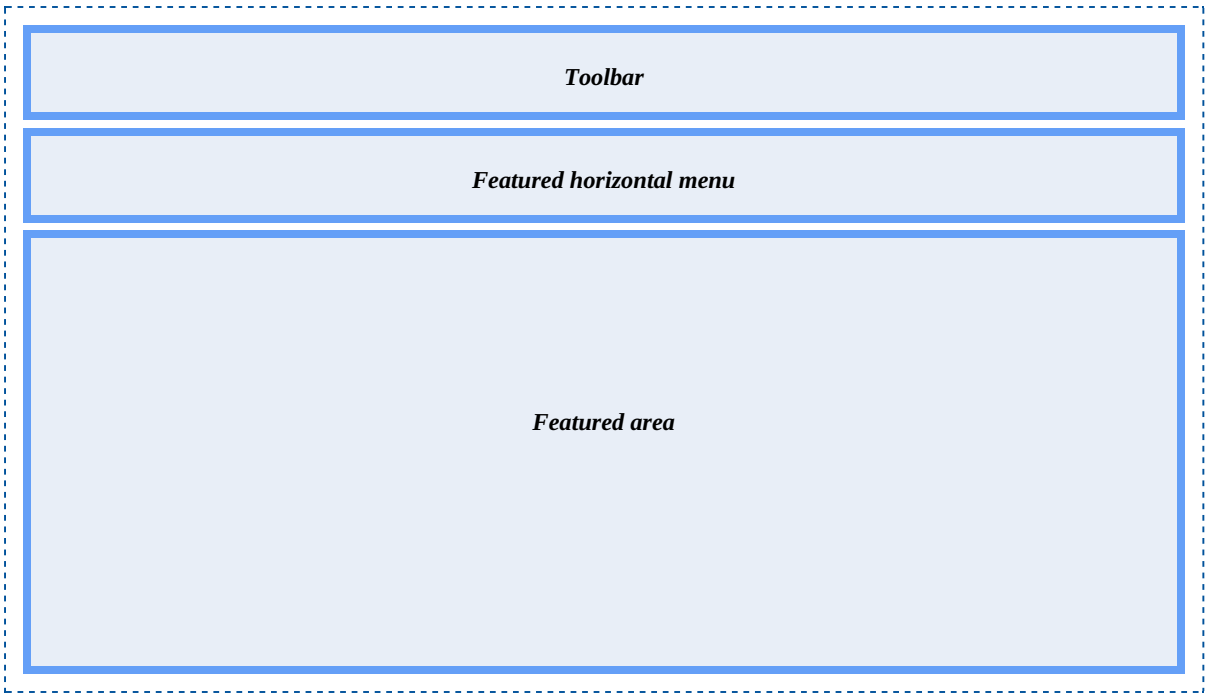
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Administration guide

OpenKM provides easy access to the application configuration by means of the "Administration Tab" located on the administration system area. Relevant and useful information can be accessed to assist the administrator in controlling and configuring the OpenKM repository.

> *Only users with **ROLE_ADMIN** can access the administration area. A new tab **Administration** will appear to these users at the right top corner of system toolbar.*

Administration layout:



OpenKM administration provides access to:

Icon	*Name*	*Description*
	Home	*Administration landing page. Here is shown:* • *Application version.* • *Installation UUID.*
	Configuration parameters	*Almost all configuration parameters are set here.* *With multi-tenant module enabled is possible to set different properties depending on the tenant.*

		Changes done here affect all the application.
	Mime types	*All registered document MIME types which OpenKM understands. You can modify, remove or add a new one.*
	Statistics	*Shows information about* ***repository size****. Also can check the evolution of the* ***Pending task queue*** *and* ***Pending extraction queue****.*
	Metadata	*Creates, modifies or removes metadata definitions.*
	Users	*Creates, modifies or removes users and roles.* *When the application is configured to connect LDAP, these values can only be changed on LDAP server because OpenKM connect to LDAP only in read only mode.*
	Profile	*A profile is a specific set of configurations for the user interface.*
	Relations	*Creates, modifies or removes node relation type.*
	User interface styles definition	*Modifies the CSS used by the user interface.*
	Reports	*Adds, removes or executes application reports.*
	Activity log	*User actions are registered in the OpenKM activity log.* *All activity log data is stored in* ***OKM_ACTIVITY*** *table.*
	Wofkflow	*Adds, removes or view status of registered workflows.*
	Automation	*Sets automatic tasks based on application events like DOCUMENT_CREATION, among others.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	Stamp	*Sets text or image stamp definitions.* *This option is hidden when the extension is not enabled. Read about it at Enable extensions.*
	Crontab	*Adds or removes scheduled crontab tasks. The application uses these tasks to perform actions like indexing documents, antivirus analyse among others.*
	OMR (Object Mark Recognition)	*Sets Object Mark Recognition parameters.*
	OCR templates	*Sets OCR template definitions to automate document?s data capture.*
	Folder styles definion	*Adds, modify or remove folder styles definitions.*
	Languages	*Add, modifies or remove languages and manage translations.*
	Utilities	*A collection of advanced utilities:* • *Admin permissions.* • *Cache Stats.* • *Check e-mail.* • *Check text extraction.* • *Clean conversion cache.* • *Database query.* • *Execute JAR.* • *Download tokens.* • *Hibernate stats.* • *LDAP search.* • *List indexes.* • *LogCat* • *Plugins* • *Profile stats.*

		• *Purge permissions.* • *Purge trashes.* • *Rebuild indexes.* • *Repository checker.* • *Repository export.* • *Repository import.* • *Scripting.* • *System properties.* • *Thesaurus.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuration parameters

OpenKM is preconfigured with a number of parameters. However, if you plan to use it in production you should take a look at Recommended configuration parameters. To access Configuration parameters go to ***Administration > Configuration parameters****.*

Configuration parameters are shown in a table with three columns:

Property Key	Type	Value

Property key is the unique identifier for each configuration property.

There are several configuration types:

- *String.*
- *Integer (only* ***numeric*** *values are allowed).*
- *Boolean (only* ***true*** *or* ***false*** *values allowed).*
- *List (each break line is considered as new list element).*
- *Text.*
- *HTML (HTML tags allowed).*
- *File.*

To create your own parameters read Creating your own parameters.

Edit a parameter

- *Click* ***edit icon*** .
- *View and* ***modify the value****.*
- *Click* ***Edit button*** *to confirm the change.*

Delete a parameter

You should not delete parameters if you're not totally sure about the implications of this action.

- *Click* ***remove icon*** .
- *In* ***Delete confirmation view*** *click* ***Delete button****.*

Backup configuration parameters

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click at top right the **Configuration parameters export icon** .*
- *Browser will download an SQL file with the configuration.*

Restore configuration parameters

Restoring configuration parameters will remove actual parameters and will be replaced by newers.

- *Go to **bottom page**.*
- *Select **configuration parameter SQL file**.*
- *Click **Import button**.*

Check configuration

This action will check if some conflictive parameters are well configured (pdf2swf, convert tool, OCR engine, OpenOffice or LibreOffice path).

Although a parameter may pass this check, it that not means it is 100% correct. Take it as a light check which cannot covers all cases but is good approximation to discard most problems.

- *Click at top right the **Check icon** .*
- *It will show the **Configuration check view**. If some parameters are misconfigured or wrong, it will be shown in red.*

Multitenant configuration

In OpenKM 7.1 multitenant mode is enabled by default.

- *Click at top right **Tenant configuration button**.*
- *In tenant configuration you can create your own configuration parameters. These will have preference over the default values. For example, have another **application.url** value for each tenant.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Recommended configuration parameters

There are some recommended changes in the configuration:

Field / Property	*Type*	*Description*
application.url	***String***	*This parameter is used in mail notifications and the copy-to-clipboard feature among others.* *By default OpenKM sets the access URL to* ***http://localhost:8080/openkm/index****, but obviously you don't want to access your OpenKM installation only from localhost.*

notification.message.body	***HTML***	*This parameter is used in mail notifications body.*
		`Message: ${notificationMessage}`
		`User: ${userId}`
		`<#list documentList as doc>Document: ${doc.path}`

notification.message.subject	***Text***	*This parameter is used in mail notifications subject. Help users identify that these emails are coming from OpenKM.* `OpenKM - NOTIFICATION`
subscription.message.enabled	***Boolean***	*Whether the suscription is evaluated and the message is sent or not. By default is enabled, but in case of massive document import we recommend to disable it to avoid performance issues.*

subscription.message.body	***HTML***	*This parameter is used in mail user subscription notification body.*
		`Date: ${eventDate?string("yyyy-MM-dd HH:mm:ss")}`
		`Document: ${documentPath}`
		`User: ${userId}`
		`Event: ${eventType}`
		`Comment: ${subscriptionComment}`

subscription.message.subject	***Text***	*This parameter is used in mail subscription notification subject. Help users identify that these emails are coming from OpenKM.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`OpenKM - ${eventType} - ${documentPath}`
text.extraction.concurrent	***Boolean***	*Enable or disable concurrent text extraction concurrent threads.*
text.extraction.batch	***Integer***	*It indicates the number of documents being processed on each indexing cycle.* *When parameter "**text.extraction.concurrent" is enabled**, is a good practice to set multiple of "**text.extraction.threads**" value.*
text.extraction.threads	***Integer***	*Number of concurrent threads that will be used for text extraction.* *It's a good practice to set this number to be less or equal than CPU cores you want to use for text extraction.*
mail.extraction.concurrent	***Boolean***	*Enable or disable concurrent mail extraction concurrent threads.*
mail.extraction.batch	***Integer***	*It indicates the number of mails being processed on each indexing cycle.* *When parameter "**mail.extraction.concurrent" is enabled**, is a good practice to set multiple of "**mail.extraction.threads**" value.*
mail.extraction.threads	***Integer***	*Number of concurrent threads that will be used for mail extraction.* *It's a good practice to set this number to be less or equal than CPU cores you want to use for mail extraction.*
system.imagemagick.convert	***String***	*Path to ImageMagick convert tool.*
system.ocr	***String***	*Path to OCR text extractor engine.*
system.swftools.pdf2swf	***String***	*Path to pd2swf conversion tool.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

system.openoffice.path	***String***	*Path to OpenOffice or LibreOffice installation.*
logo.login	***File***	*Login page logo.* *It automatically tries to resize image to 316px × 74px (suggested background #F1F3F5).*
logo.mobile	***File***	*Mobile login page logo.* *It automatically tries to resize image to 161px × 38px (suggested background #F1F3F5).*
logo.report	***File***	*Default logo used in Jasper Reports.* *It automatically tries to resize image to 150px × 35px (suggested background #FFF000).*
logo.tiny	***File***	*Small logo at top right corner in frontend UI.* *It automatically tries to resize image to 51px × 13px (suggested background transparent).*
logo.favicon	***File***	*Small icon in left side of browser url.* *The uploaded file must be a valid icon (16px x 16px).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Preview parameters

In this section you can find the most common configuration parameters related to preview. Some of these commands are included in OpenKM by default, like ***pdf2swf*** *or* ***ffmpeg*** *(binaries compiles for Linux and Windows) but others need to be installed by the system administrator.*

Linux

Field / Property	*Type*	*Description*
system.imagemagick.convert	***String***	*Path to ImageMagick convert tool.* `/usr/bin/convert`
system.ocr	***String***	*Path to OCR text extractor engine.* `/usr/bin/tesseract ${fileIn} ${fileOut}`
system.ffmpeg	***String***	*Path to FFmpeg audio and video conversion tool.* `/home/openkm/tomcat-7.0.27/bin/ffmpeg`
system.html2pdf	***String***	*Path to html2pdf conversion tool. You can download this utility from http://phantomjs.org.* `/home/openkm/phantomjs /home/openkm/rasterize.js ${urlIn} ${fileOut} A4` *Other tools:*

Tool	*Description*
cutycapt	*Sometimes we have detected better results using http://cutycapt.sourceforge.net/, which seems more configurable. This utility requires an X server, so if you are running from a non-graphical environment you may need to install a virtual X server like xvfb:* `/usr/bin/xvfb-run /usr/bin/cutycapt --url=${urlIn} --out=${fileOut}`

		wkhtmltopdf `/usr/bin/wkhtmltopdf ${urlIn} ${fileOut}`
system.office2pdf	***String***	*Command line used to make convertion from Office documents to PDF, in special in* ***OfficeUnoToPdfConversion****. If you are using this conversor, LibreOffice is not required to start as a service so you can avoid it by setting the* ***system.openoffice.path*** *configuration property to an empty value.* `/usr/bin/unoconv -o ${fileOut} ${fileIn}`
system.pdfimages	***String***	*Path to pdfimages extractor tool.* `/usr/bin/pdfimages -j -f ${firstPage} -l ${lastPage} ${fileIn} ${imageRoot}`
system.openoffice.path	***String***	*Path to OpenOffice or LibreOffice installation path.* `/usr/lib/libreoffice`
system.openoffice.program	***String***	*Path to OpenOffice or LibreOffice executable. Used in* ***OfficeDirectToPdfConversion****. If you are using this conversor, LibreOffice is not required to start as a service so you can avoid it by setting the* ***system.openoffice.path*** *configuration property to an empty value and set* ***system.openoffice.server*** *to "local".* `/usr/bin/soffice`
system.ghostscript	***String***	*Path tot the ghosscript conversion tool.* `/usr/bin/gs`

Windows

Field / Property	*Type*	*Description*
system.imagemagick.convert	***String***	*Path to image magick convert tool.* `C:\openkm\bin\convert.exe`
system.ocr	***String***	*Path to ocr text extractor engine.*

		`C:\openkm\tools\Tesseract-OCR\tesseractexe ${fileIn} ${fileOut}`
system.ffmpeg	***String***	*Path to ffmpeg audio and video converter tool.* `C:\openkm\bin\ffmpeg.exe`
system.html2pdf	***String***	*Path to html2pdf convert tool. You can download this utility from http://phantomjs.org.* `C:\openkm\tools\phantomjs C:\openkm\tools\rasterize.js ${urlIn} ${fileOut} A4`
system.pdfimages	***String***	*Path to pdfimages extractor tool.* `C:\openkm\tools\pdfimages -j -f ${firstPage} -l ${lastPage} ${fileIn} ${imageRoot}`
system.openoffice.path	***String***	*Path to Open Office or Libre Office installation path.* `C:\Program Files\LibreOffice`
system.openoffice.program	***String***	*Path to OpenOffice or LibreOffice executable. Used in* ***OfficeDirectToPdfConversion****. If you are using this conversor, LibreOffice is not required to start as a service so you can avoid it by setting the* ***system.openoffice.path*** *configuration property to an empty value and set* ***system.openoffice.server*** *to "local".* `C:\Program Files\LibreOffice\program\soffice.exe`
system.ghostscript	***String***	*Path to the ghosscript conversion tool.* `C:\openkm\bin\gswin32c.exe`

LDAP configuration parameters

The OpenKM LDAP integration is based on a couple of LDAP searches to retrieve data.

LDAP queries are composed by tree elements:

1. *Attribute or object to retrieve.*
2. *Node base to scope the request.*
3. *Filtering options.*

Sample query

The query to retrieve mail attribute from an object of type person with sAMAccountName attribute value with {0} - this parameter indicate the userId - from node cn=users,dc=company,dc=local and descendants.

principal.ldap.mail.attribute	`mail`
principal.ldap.mail.search.base	`cn=users,dc=company,dc=local`
principal.ldap.mail.search.filter	`(&(objectclass=person)(sAMAccountName={0}))`

LDAP queries

- ***principal.ldap.user.**** *are used to retrieve user list.*
- ***principal.ldap.username.**** *are used to retrieve user name.*
- ***principal.ldap.mail.**** *are used to retrieve user mail adress.*
- ***principal.ldap.role.**** *are used to retrieve role list.*
- ***principal.ldap.roles.by.user.**** *are used to retrieve the roles of a user.*
- ***principal.ldap.users.by.role.**** *are used to retrieve users of a role.*

For retrieving data, are injected arguments into option filters. For example, for getting the user mail, the application use parameter {0} to set argument value - userId - into filtering options.

Queries that use parameters to filtering:

- ***principal.ldap.mail.search.filter**=(sAMAccountName=**{0}**). Where {0} is attribute value retrieved from principal.ldap.user.attribute.*
- ***principal.ldap.roles.by.user.search.filter**=(&(objectclass=user)(sAMAccountName={0})). Where **{0}** is attribute value retrieved from principal.ldap.user.attribute.*
- ***principal.ldap.username.search.filter**=(sAMAccountName={0}). Where **{0}** is attribute value retrieved from principal.ldap.user.attribute.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***principal.ldap.users.by.role.search.filter***=*(&(objectClass=group)(cn={0})). Where* ***{0}*** *is attribute value retrieved from principal.ldap.role.attribute.*

Parameters

Field / Property	*Type*	*Description*
principal.adapter	***String***	*Class used to connect with LDAP.*
system.login.lowercase	***String***	*By default is empty. In case connecting to AD (Microsoft Active Directory) must be set to "**true**", that force all users to be logged with lowercase user Id. The reason is OpenKM is case sensitive and Microsoft Active Directory not.*
principal.ldap.server	***String***	*LDAP server.*
principal.ldap.security.principal	***String***	*LDAP user distinguished name (dn).*
principal.ldap.security.credentials	***String***	*LDAP user password.*
		This property specifies how the referrals sent by AD in the search results are handled by OpenKM. In almost most cases this value is empty. ***Note for Active Directory (AD) users:*** *AD servers are apparently unable to handle referrals automatically, which causes a* `PartialResultException` *to be thrown whenever a referral is encountered in a search. To avoid this, set the* `ignorePartialResultException` *property to* `true`*. There is currently no way of manually handling these referrals in the form of* `ReferralException`*, i.e. either you get the exception (and your results are lost) or all referrals are ignored (if the server is unable to handle them properly. Neither is there any simple way to get notified that a* `PartialResultException` *has been ignored (other than in the log).* *More information at Spring LdapTemplate.* *Values might be:*

principal.ldap.referral	***String***	*"**ignore**"* *If principal.ldap.`referral=ignore`, the following exception will be thrown when a referral is encountered:* *javax.naming.PartialResultException: Unprocessed Continuation Reference(s); remaining name 'dc=company,dc=com'* *"**follow**"* *If `principal.ldap.referral=follow`, OpenKM will automatically follow the referral. To be successful, make sure OpenKM can access the referred server. The errors about ReferralExceptions should not be shown.* *"**throw**"* *If principal.`ldap.referral=throw`, the following exception will be thrown:* *com.sun.jndi.ldap.LdapReferralException: Continuation Reference; remaining name 'dc=company,dc=com'* *More information at Referrals in the JNDI.*
principal.ldap.users.from.roles	***Boolean***	*When "**true**" tries to retrieve all users from roles list.*
principal.ldap.user.attribute	***String***	*User attribute.*
principal.ldap.user.search.base	***List***	*List of nodes base.*
principal.ldap.user.search.filter	***String***	*Filter options.*
principal.ldap.username.attribute	***String***	*Username attribute*
principal.ldap.username.search.base	***String***	*Node base.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.username.search.filter	***String***	*Filter options.*
principal.ldap.mail.attribute	***String***	*Mail attribute.*
principal.ldap.mail.search.base	***String***	*Node base.*
principal.ldap.mail.search.filter	***String***	*Filter options.*
principal.ldap.role.attribute	***String***	*Role attribute.*
principal.ldap.role.search.base	***List***	*List of nodes base.*
principal.ldap.role.search.filter	***String***	*Filter options.*
principal.ldap.roles.by.user.attribute	***String***	*User attribute.*
principal.ldap.roles.by.user.search.base	***String***	*Node Base.*
principal.ldap.roles.by.user.search.filter	***String***	*Filter options.*
principal.ldap.users.by.role.attribute	***String***	*Role attribute.*
principal.ldap.users.by.role.search.base	***String***	*Node base.*
principal.ldap.users.by.role.search.filter	***String***	*Filter options.*

Sample configuration

Field / Property	*Type*	*Description*
principal.adapter	***String***	`com.openkm.plugin.principal.LdapPrincipalAdapt`
system.login.lowercase	***String***	`true`
principal.ldap.server	***String***	`ldap://192.168.1.20:389`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.security.principal	***String***	`CN=Administrator,OU=OpenKM,DC=company,DC=com`
principal.ldap.security.credentials	***String***	`password`
principal.ldap.referral	***String***	
principal.ldap.users.from.roles	***Boolean***	`false`
principal.ldap.user.attribute	***String***	`sAMAccountName`
principal.ldap.user.search.base	***List***	`DC=company,DC=com`
principal.ldap.user.search.filter	***String***	`objectclass=person`
principal.ldap.username.attribute	***String***	`cn`
principal.ldap.username.search.base	***String***	`DC=company,DC=com`
principal.ldap.username.search.filter	***String***	`(sAMAccountName={0}`
principal.ldap.mail.attribute	***String***	`mail`
principal.ldap.mail.search.base	***String***	`DC=company,DC=com`
principal.ldap.mail.search.filter	***String***	`(sAMAccountName={0}`
principal.ldap.role.attribute	***String***	`cn`
principal.ldap.role.search.base	***List***	`DC=company,DC=com`
principal.ldap.role.search.filter	***String***	`objectclass=group`
principal.ldap.roles.by.user.attribute	***String***	`memberOf`
principal.ldap.roles.by.user.search.base	***String***	`DC=company,DC=com`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

principal.ldap.roles.by.user.search.filter	***String***	`(&(objectclass=person)(sAMAccountName={0}))`
principal.ldap.users.by.role.attribute	***String***	`member`
principal.ldap.users.by.role.search.base	***String***	`DC=company,DC=com`
principal.ldap.users.by.role.search.filter	***String***	`(&(objectClass=group)(cn={0}))`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

User interface configuration parameters

The uploaded images - except in case of the favicon - are automatically resized to match the right size. Most image formats like jpg, png, etc are allowed .

Field / Property	Type	Description
default.ui.context	**String**	*The context of the application. By default OpenKM comes with 2 users interfaces, the classic UI and kcenter.* *Values are:* ** For classic UI use "**/frontend/index**"* ** For kcenter UI use "**/kcenter/**"* `/kcenter/`
default.lang	**String**	*Forces user interface language to a specific language. By default this parameter is empty. The value is based on international language code.* `en-GB`
security.mode.multiple	**Boolean**	*Enabled alternative security mode that allows to apply several changes at same time. Only one change is applied each time. By default the value is "false".* *After the parameter is changed, refresh the browser to take effect.* `false`
logo.favicon	**String**	*Website icon.*
logo.login	**String**	*Login page logo.* *Automatically it tries to resize image to 316px × 74px (suggested background #F1F3F5).*
logo.mobile	**String**	*Mobile logo.*

		Automatically it tries to resize image to 161px × 38px (suggested background #F1F3F5).
logo.report	***String***	*Reports logo.* *Automatically it tries to resize image to 150px × 35px (suggested background #FFF000).*
logo.tiny	***String***	*A small logo is used at the top right in OpenKM user interface.*
text.banner	***String***	*Text shown at top of the logo in the landing page.* *Automatically it tries to resize image to 51px x 13px (suggested background #E5E5E1).*
text.title	***String***	*Browser title. This text is set between* `${text.title}` *html tags.*

text.welcome	***String***	*Text shown in the login table at the landing page.*
		`Welcome to OpenKM !`
		`Use a valid username and password to access to OpenKM user Desktop.`

thumbnail.size.lightbox	***String***	*Maximum size - width * height - of the lightbox thumbnail.* `640x640>`
thumbnail.size.properties	***String***	*Maximum size - width x height - of the thumbnail in properties view.* `280x280>`
thumbnail.size.search	***String***	*Maximum size - width x height - of the thumbnail in search view.* `180x180`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

keep.session.alive.interval	***Integer***	*Some administration tasks take a long time, like repository checker and rebuild Lucene indexes, these can cause the user session to be lost. For this reason these type of operations are forced to maintain the session alive. By default is set to 5 minutes.* `5`
schedule.session.keepalive	***Integer***	*It is used by the user interface to maintain the browser connected to OpenKM even when a user stays for some time without performing any operation. For this reason, it should never be lower than server session timeout (by default set to 15 minutes). By default is set to 5 minutes.* `5`
schedule.dashboard.refresh	***Integer***	*The application periodically does some operations like refreshing the dashboard panel to get new imported mails, getting repository news or showing latest documents created by other users among others. The time set the period between dashboard panel will be refresh.* `15`
schedule.ui.notification	***Integer***	*User receives notification with a popup. The time - in minutes - indicate the period used by application looking for new messages in server.* `1`
system.previewer	***Select***	*The application can use "flexpaper" or "zviewer" tool to preview documents. By default flexpaper previewer is enabled.* `flexpaper`
download.zip.live.max.files	***Integer***	*Sets the maximum number of files allowed to download in live mode. Once this number is exceeded,the application will notify the user that the ZIP creation will be executed on background mode. And after finished will notify the user - by mail - with a download URL. Value **0** indicate **unlimited number of files**.* *Downloading a couple of selected files and folders can take a lot of time. The application on the server side creates a temp folder where all data is packed as a ZIP file and then*

		is forwarded to the end user interface. `100`
download.zip.live.max.file.size	***String***	*Sets the maximum size of all files allowed to download in live mode. Exceeded this size, the application will notify the user that the the ZIP creation will be executed on background mode. After it is finished, it will notice the user - by mail - with a download URL. Value* ***0*** *indicate* ***unlimited size****. Indicates size units with KB, MB, GB or TB, otherwise value will be considered as bytes.* *Downloading a couple of selected files and folders can take a lot of time. The application on the server side creates a temp folder where all data is packed as a ZIP file and then is forwarded to the end user interface.* `100MB`
min.search.characters	***Integer***	*Sets the minimum characters required to execute a search. With less characters the search button will not be enabled.* `3`
custom.header.height	***Integer***	*Sets the height reserved at the top of the user interface for a customized horizontal banner.* `0`
custom.header.source	***String***	*Sets the relative path of the custom header. When this parameter is not empty and parameter "****custom.header.height****" value is greater than 0, will be shown an horizontal section at top of the user interface with the content set in this url.* `/OpenKM/custom/header.jsp`
custom.login	***String***	*Changes default login screen with your own.* `/custom/login.jsp`
force.reset.chat.when.login	***Boolean***	*When enabled force to logout the chat when loading. This parameter it's*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		useful for users who are not logout correctly the application and leaving chat session alive or when users are working with the application in several browser tabs at the same time. `true`
ask.drag.and.drop.updates	***Boolean***	*Deprecated parameter. From OpenKM version 7.1.21, it is replaced by a configuration parameter in the Profiles.* *Ask user in a drag and drop operation if existing files should be updated.* `false`
drag.and.drop.enabled	***Boolean***	*This parameter can be used to enable or disable drag and drop in file browser.* `true`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow configuration parameters

Explain for what these parameteres are:

Field / Property	*Type*	*Description*
workflow.adapter	***String***	*OpenKM can be used with several workflow engines. By default, we offer an integration with jBPM but by implementing an adapter you can use other workflow engines. To do this, you need to implement the* ***com.openkm.workflow.adapter.WorkflowAdapter*** *interface.* *Possible values* • *com.openkm.workflow.adapter.DummyWorkflowAdapter* • *com.openkm.workflow.adapter.JbpmRemoteWorkflowAdapter* `com.openkm.workflow.adapter.DummyWorkflowAdapter` *When DummyWorkflowAdapter is selected the workflow icon in administration will not being shown.* *After changing this value you should restart OpenKM application to take effect.*
workflow.adapter.login	***String***	*Login credentials used to connect to the external workflow engine. Used with* ***com.openkm.workflow.adapter.JbpmRemoteWorkflowAdapter****.* `admin`
workflow.adapter.password	***String***	*Password used to connect to the external workflow engine. Used with* ***com.openkm.workflow.adapter.JbpmRemoteWorkflowAdapter****.* `admin`
workflow.adapter.url	***String***	*The URL used to connect to the external workflow engine. Used with* ***com.openkm.workflow.adapter.JbpmRemoteWorkflowAdapter****.* `http://localhost:8080/jbpm-console`
workflow.run.config.form	***String***	*The form name in case the workflow needs parameters at the start. By default this form is called "run_config".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`run_config`
workflow.start.task.auto.run	***String***	*If OpenKM should run the task defined in the start node. This start dummy task is usually used to assign the "initiator" swimlane. By default the value is set to "true". You don't need to modify this value.* `true`
workflow.node.deletion.check	***Boolean***	*Enable or disable the deletion of a node which is used in a workflow. By default is set to "true".* `true`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Notification configuration parameters

When a user is subscribed to a document, any change in the document will be notified to by email. These email messages are created using some defined templates. These default templates to create messages can be adapted to your own style.

Field / Property	Type	
send.mail.from.user	**Boolean**	*By default the application sends mail notifications setting the from mail field valu mails sent by the application (all mails sent by the application will have the sam* *When mail server have restricted to change the default from mail fie value user@mail.com , usually this kind of mail server configuration* `true`
notification.message.subject	**Text**	*Subject of the message sent when a user has chosen notify something to other.* `OpenKM - NOTIFICATION`

notification.message.body	**HTML**	*Body of the message sent when a user has chosen notify something to other.*
		 `ReplaceMessage: ${notificationMessage}`
		 `User: ${userId}`
		 `<#list documentList as doc>Document: ${doc.path}`

subscription.message.subject	**Text**	*Subject of the message sent automatically caused by some event. That happens b* `OpenKM - ${eventType} - ${documentPath}`

subscription.message.body	**HTML**	*Body of the message sent automatically caused by some event. That happens bec*
		 `Date: ${eventDate?string("yyyy-MM-dd HH:mm:ss")}`
		 `Document: ${documentPath}`
		 `User: ${userId}`
		 `Event: ${eventType}`
		 `Comment: ${subscriptionComment}`

task.message.subject	**Text**	*Subject of the message sent when there's some change on task assigned to a user.* `OpenKM - Task Manager Notification: ${message}`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

task.message.body	***HTML***	*Body of the message sent when there's some change on task assigned to a user.* ***Task:*** *${tasksubject}* ***Start:*** *${taskstart!}* ***End:*** *${taskend!}* ***Project:*** *${taskproject!}* ***Status:*** *${taskstatus!}* ***Note:*** *${tasknote!}*
notify.twitter.status	***Text***	*This is the text used in the notification twitter message.* *OpenKM - ${documentUrl} - ${documentPath} - ${userId}*
notify.twitter.user	***String***	*Twitter account username.*
notify.twitter.password	***String***	*Twitter account password.*
download.message.body	***HTML***	*Body of the message sent when a user share some document to other for downloa* *Your archive is ready. Please download from ${url}*
download.message.subject	***String***	*Subject of the message sent when a user share some document to other for down* *OpenKM - DOWNLOAD*
download.token.validity	***Integer***	*Range of days while the download url will be available.* *7*
template.message.body	***String***	*Set the basic template for all notifications.* *<div style="margin: 0; padding: 0; width: 100%; min-* *<table style="height: 100%; width: 100%;" border="0"* *<tbody>* *<tr>* *<td style="background: #e5e5e5;" valign="top">* *<table style="margin: 20px auto 0 auto; width: 600px* *<tbody>* *<tr style="background: #e5e5e5;">* *<td valign="top">* * * *<table width="100%" cellspacing="0" cellpadding="0">* *<tbody>* *<tr>* *<td>*

```
<table width="100%" cellspacing="0" cellpadding="0">
<tbody>
<tr>
<td bgcolor="#018ca9" width="100%" height="6px">&nbs
</tr>
</tbody>
</table>
</td>
</tr>
<tr>
<td>
<div style="border-top: solid 1px #88888b; min-heigh
</td>
</tr>
<tr>
<td style="padding: 0 10px 10px 10px;">
<div>
<p>${body}</p>
</div>
</td>
</tr>
</tbody>
</table>
</div>
</td>
</tr>
<tr>
<td style="background: #88888b;">
<table style="width: 100%;" border="0" cellspacing="
<tbody>
<tr>
<td height="4"> </td>
</tr>
</tbody>
</table>
</td>
</tr>
<tr>
<td style="background: #222222; padding: 10px 20px;"
<table style="width: 100%;" border="0" cellspacing="
<tbody>
<tr>
<td>
<h3 style="color: #f6a81a; font: 18px verdana, arial
</td>
</tr>
</tbody>
</table>
</td>
</tr>
</tbody>
</table>
</td>
</tr>
</tbody>
</table>
</div>
```

notification.url	***String***	*The notification URL parameter is the template URL what is used by OpenKM fo* *Classic user interface:* `http://localhost:8080/openkm/index`

Kcenter user interface:

```
http://localhost:8080/openkm/kcenter/
```

Substitutions

*For **notification.message.body** property:*

- ***${documentUrl}** shows the node URL..*
- ***${documentPath}** shows the node path.*
- ***${documentName}** show the node name.*
- ***${userName}** show the user name who caused the event.*
- ***${userId}** show the user id who caused the action.*
- ***${notificationMessage}** show the notification message*

*For **task.message.body** property:*

- ***${tasksubject}** show the subject of the task.*
- ***${taskstart}** show when task started.*
- ***${taskend}** show when task ended.*
- ***${taskproject}** the task project.*
- ***${taskstatus}** indicate the status of the task.*
- ***${tasknote}** show the notes.*

*FreeMarker does not allow you to render null values (will throw an exception). If you expect a value to be null you can also use the **!** operator: **${content.title!}**. This is the equivalent to **${content.title!}**.*

More information at:

- *java.text.MessageFormat*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Document version numeration configuration

We support different numeration schemas. By default OpenKM uses ***com.openkm.vernum.MajorMinorVersionNumerationAdapter*** *which is similar to Jackrabbit implementation.*

More information about how to Creating your own Version Number Adapter.

Field / Property	*Type*	*Description*
version.numeration.adapter	***String***	*Numeration schema.* *Available schemas :* • *com.openkm.plugin.vernum.PlainVersionNumerationAdapt* • *com.openkm.plugin.vernum.PlainZeroVersionNumerationA* • *com.openkm.plugin.vernum.MajorMinorVersionNumeratio* • *com.openkm.plugin.vernum.BranchVersionNumerationAda* • *com.openkm.plugin.vernum.MajorMinorReleaseVersionNu* • *com.openkm.plugin.vernum.VersionLetterAdapter* `com.openkm.vernum.MajorMinorVersionNumerationAdapter`
version.numeration.format	***String***	*Numeration number format. Used in* ***PlainVersionNumerationAdapter****.* `%d`

Some Numeration Adapters like ***PlainVersionNumerationAdapter*** *may have additional configuration parameters like* ***version.numeration.format*** *which is used to define the numeration number format. By default this is "%d" but you can change to "%02d" to create versions like "01", "02", etc.*

Restarting application *is needed after changing the value of* ***version.numeration.adapter.***

Also note that if already there are documents in the repository, changing the Numeration Adapter may cause problems because they are not compatible between them.

Version numeration schemas

Numeration adapter	*First version*	*Next versions*
com.openkm.plugin.vernum.PlainVersionNumerationAdapter	*1*	*2, 3, 4, etc.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

com.openkm.plugin.vernum.PlainZeroVersionNumerationAdapter	*0*	*1, 2, 3, etc.*
com.openkm.plugin.vernum.MajorMinorVersionNumerationAdapter	*1.0*	*1.1, 1.2, 1.3, etc.*
com.openkm.plugin.vernum.BranchVersionNumerationAdapter	*1*	*2, 3, 2.1, 2.2, etc.*
com.openkm.plugin.vernum.MajorMinorReleaseVersionNumerationAdapter	*0.0.0*	*0.0.1, 0.0.2, 0.1.0, 0.1.1, 1.0.0*
com.openkm.plugin.vernum.VersionLetterAdapter	*A*	*B, C*

Migration between versions

***We recommend you set the correct numeration at the beginning of using the application** . Otherwise will be needed a migration of existing numeration and this operation is not always easy.*

*You can change the default **Numeration Adapter** using the "**version.numeration.adapter**" configuration parameter. You need to restart OpenKM every time you change this parameter. Also note that if already there are documents in the repository, changing the **Numeration Adapter** may cause problems when they are not compatible between them.*

*If you change the version numeration adapter you also need to update the existing numeration, for example the migration from **PlainVersionNumerationAdapter** to **MajorMinorVersionNumerationAdapter**.*

```
-- HQL queries
UPDATE NodeDocumentVersion ndv SET ndv.name = '01' WHERE ndv.name='1.0';
UPDATE NodeDocumentVersion ndv SET ndv.name = '02' WHERE ndv.name='1.1';
UPDATE NodeDocumentVersion ndv SET ndv.name = '03' WHERE ndv.name='1.2';
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Password validation configuration

Validation password enables the application to set a couple of rules that should be accomplished by passwords. By default, application makes no password checking (***validator.password*** *property is set by default to* ***com.openkm.validator.password.NoPasswordValidator*** *).*

To enable validator password must change to ***com.openkm.validator.password.CompletePasswordValidator*** *or Creating your own password validator.*

Field / Property	*Type*	*Description*
validator.password	***String***	*Description* *Available:* • *com.openkm.validator.password.NoPasswordValidator* • *com.openkm.validator.password.CompletePasswordValidator* `com.openkm.validator.password.NoPasswordValidator`
validator.password.min.length	***String***	*Minimum length allowed.* `Numeric value greater than 0.`
validator.password.max.length	***String***	*Maximum length allowed.* `Numeric value greater than 0.`
validator.password.min.lowercase	***String***	*Minimum number of lower-case characters.* `Numeric value greater than 0.`
validator.password.min.uppercase	***String***	*Minimum number of upper-case characters.* `Numeric value greater than 0.`
validator.password.min.digits	***String***	*Minimum number of digits.* `Numeric value greater than 0.`
validator.password.mini.special	***String***	*Minimum number of special characters.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`Numeric value greater than 0.`

Sample configuration

- *Must have at least 8 characters.*
- *At least must have 1 upper-case character.*
- *At least must have 1 digit.*

Field / Property	*Type*	*Description*
validator.password	***String***	`com.openkm.validator.password.CompletePasswordValid`
validator.password.min.length	***String***	`8`
validator.password.max.length	***String***	
validator.password.min.lowercase	***String***	
validator.password.min.uppercase	***String***	`1`
validator.password.min.digits	***String***	`1`
validator.password.mini.special	***String***	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Switch application to read-only mode

It can force the whole repository to be in read-only mode. This is useful when doing maintenance tasks, or while migrating and you wish to get the system alive for your users but want to ensure nobody can change anything.

Field / Property	*Type*	*Description*
system.readonly	*Boolean*	*Switch whole repository to be read-only. By default is set to false.* `false`

*If you **restart the application**, this parameter will be **automatically set to true**.*

*For security and performance reasons, when you execute **Lucene rebuild indexes**, the system will **switch automatically to read-only mode** until the process is finished. Do not change manually, wait until be completed.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Security configuration parameters

Field / Property	*Type*	*Description*
user.admin	***String***	*Sets the default super user id. By default super user id is okmAdmin.* `okmAdmin`
default.user.role	***String***	*Sets the default general connection role.* *Any user to successfully login in the screen needs to have **default.user.role** or **default.admin.role**, otherwise will get 403 error.* `ROLE_USER`
default.admin.role	***String***	*Sets the default administration connection role. This role will get users access to Administration tools.* *Any user to successfully login in the screen need to have **default.user.role** or **default.admin.role**, otherwise will get 403 error.* `ROLE_ADMIN`
user.assign.document.creation	***Boolean***	*By default when a user creates a node, he is added to the node with full permissions. You can disable this behavior by setting this parameter to false. By default the parameters is set to true.* `true`
user.password.expiration	***Integer***	*Indicates the number of days to force user change their password, the 0 value indicates the feature is disabled.* `0`
user.password.reset	***Boolean***	*Sometimes the user loses or forgets their password. When this option is enabled, used in landing page have got the option to restart their password and be notified with a newer one by mail. By default this option is set to false.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`false`
system.antivir	***String***	*Path to antivirus.* *An antivirus can decrease perfomance, taking several seconds to analyze each document and consuming a lot of CPU resources.*
security.access.manager	***String***	*Sets security access manager determinate the way on how the security is evaluated. Default value is "simple".* *Values:* • *simple.* • *recursive.* • *read_recursive.* `simple` *Takes a look at Security configuration*
security.search.evaluation	***String***	*Sets the security search manager and determinates the way how the security is evaluated by search engine.* *Values:* • *lucene.* • *am_more, am_window or am_limited.* *Take a look at Security configuration*
security.extended.mask	***Integer***	*Security evaluation can be extended to downloaded files, starts workflows, adds, removes or modifies property groups (metadata) or compact history. Default value is empty.* *Values:* • *DOWNLOAD = 1024;* • *START_WORKFLOW = 2048;* • *COMPACT_HISTORY = 4096;*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		• *PROPERTY_GROUP = 8192;* *For example to enable download and property groups you should put the mask 9216 (1024+8192)* `9216` *Take a look at Security configuration*
security.live.change.node.limit	***Integer***	*When security changes affect more than the values set in this property the security changes are applied as a background task.* `100` *Take a look at Security configuration*
system.login.lowercase	***Boolean***	*By default is empty. In case connecting to AD (Microsoft Active Directory) must be set to "**true**", that force all users to be logged with lowercase user Id. The reason is OpenKM is case sensitive and Microsoft Active Directory not.*
principal.adapter	***String***	*You must restart OpenKM service after you change this parameter.* *OpenKM can handle user access using the Spring Security framework. OpenKM needs an available method for reading users and roles, so when users are stored in a database (as is the default), the class **DatabasePrincipalAdapter** does this job.* `com.openkm.core.DatabasePrincipalAdapter` *If you configure OpenKM to authenticate against an LDAP server, you need to configure another principal adapter like **LdapPrincipalAdapter**.* *More information at:* • *LDAP configuration.* • *Creating your own login module.*

principal.identifier.validation	***String***	*Sets a regular expression to validate on the creation of user names and roles names.* `^[a-zA-Z0-9_]+$` *This parameter only can be used in combination with "**principal.adapter**" parameter value "**com.openkm.core.DatabasePrincipalAdapter**".*
browser.password.autocompletion	***Boolean***	*By default it is true (allowed autocompletion). This paremeter allows to disable the autocompletion of the password in the login frame. This is a security messure in not trusted clients.*
user.password.remember	***Boolean***	*By default it is false. When it is enabled a checkbox appears in the login form where you can decide if you are going to be remembered or not. This means that you won't be asked for user and password next time you access OpenKM, unless you make a logout from File > Logout.* *This configuration does not work with LDAP integration. Anyway, if yor OpenKM is configure with LDAP integration, take a look at LDAP troubleshooting which contains information about solving this incompatibility.*
default.security.recursive.role	***String***	*Set the role what is used to identify what users are able to set recursive security.* `ROLE_SECURITY_RECURSIVE`
secure.file.delete	***Boolean***	*Enable safe file delete from the file system. The process ensures the document might be recovered later from the file system.* `false`
security.login.failed.attempts	***Integer***	*Number of login failure attemps before the user will be locked. Value 0 indicates unlimited failure attemps and the user will never be locked.* `0`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

default.task.manager.admin.role	***String***	*Set the role what is used to identify what users are able to manage all the OpenKM task.* `ROLE_TASK_MANAGER_ADMIN`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Extensions configuration parameters

HTML Editor 4

The application comes with integrated WYSIWYG editor based on tinyMCE v4. The components of the editor can be parametrized with these parameters:

Field / Property	*Type*	*Description*
extension.tinymce4.plugins	***String***	*Set enabled tinyMCE plugins.* *Values:* • *advlist* • *autolink* • *lists* • *link* • *image* • *charmap* • *print* • *preview* • *hr* • *anchor* • *pagebreak* • *searchreplace* • *wordcount* • *visualblocks* • *visualchars* • *fullscreen* • *insertdatetime* • *media* • *nonbreaking* • *save* • *table* • *contextmenu* • *directionality* • *emoticons*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		• *template* • *paste* • *textcolor* • *code* `advlist autolink lists link image charmap print preview hr anchor pagebreak searchreplace wordcount visualblocks visualchars fullscreen insertdatetime media nonbreaking save table contextmenu directionality emoticons template paste textcolor code`
extension.tinymce4.theme	***String***	*Set tinyMCE theme.* `modern`
extension.tinymce4.toolbar1	***String***	*Set elements present in first toolbar.* *Values:* • *insertfile* • *undo* • *redo* • *styleselect* • *bold* • *italic* • *alignleft* • *aligncenter* • *alignright* • *alignjustify* • *bullist* • *numlist* • *outdent* • *indent* • *link* • *image* `insertfile undo redo | styleselect | bold italic |`

		`alignleft aligncenter alignright alignjustify | bullist numlist outdent indent | link image`
extension.tinymce4.toolbar2	***String***	*Set elements present in second toolbar.* *Values:* • *okm_checkin* • *okm_cancelcheckout* • *okm_searchDocument* • *okm_searchFolder* • *okm_searchImage* • *okm_codeHighlight* • *print* • *preview* • *media* • *forecolor* • *backcolor* • *emoticons* `okm_checkin okm_cancelcheckout okm_searchDocument okm_searchFolder okm_searchImage okm_codeHighlight | print preview media | forecolor backcolor emoticons`

HTML Email editor

Field / Property	*Type*	*Description*
mail.editor.tinymce4.toolbar1	***String***	*Set elements present in first toolbar.* *Values:* • *insertfile* • *undo* • *redo* • *styleselect* • *bold* • *italic*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		• *alignleft* • *aligncenter* • *alignright* • *alignjustify* • *bullist* • *numlist* • *outdent* • *indent* • *link* • *image* `okm_sendMail, okm_cancelSendMail, okm_searchDocument, okm_searchFolder,	,styleselect, |, bold, italic, |, alignleft, aligncenter, alignright, alignjustify,	,bullist, numlist, outdent, indent,	forecolor, backcolor, emoticons`

HTML task manager

Field / Property	*Type*	*Description*
taskmanager.tinymce4.toolbar1	***String***	*Set elements present in first toolbar.* *Values:* • *insertfile* • *undo* • *redo* • *styleselect* • *bold* • *italic* • *alignleft* • *aligncenter* • *alignright* • *alignjustify* • *bullist*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		• *numlist* • *outdent* • *indent* • *link* • *image* `okm_sendMail, okm_cancelSendMail, okm_searchDocument, okm_searchFolder, | ,styleselect, |, bold, italic, |, alignleft, aligncenter, alignright, alignjustify, | ,bullist, numlist, outdent, indent, | forecolor, backcolor, emoticons`

HTML Editor 3

The application comes with integrated WYSIWYG editor based on tinyMCE v3.

HTML Editor 3 is considered as deprecated and won't be longer supported. We encourage using HTML Editor version 4 instead of using version 3.

The components of the editor can be parametrized with these parameters:

Field / Property	*Type*	*Description*
		Set enabled tinyMCE plugins. *Values:* • *codehighlighting* • *autolink* • *lists* • *pagebreak* • *style* • *layer* • *table* • *save* • *advhr* • *advimage* • *advlink*

extension.tinymce.plugins	***String***	• *emotions* • *iespell* • *inlinepopups* • *insertdatetime* • *preview* • *media* • *searchreplace* • *print* • *contextmenu* • *paste* • *directionality* • *fullscreen* • *noneditable* • *visualchars* • *nonbreaking* • *xhtmlxtras* • *template* • *advlist* `codehighlighting, autolink, lists, pagebreak, style, layer, table, save, advhr, advimage, advlink, emotions, iespell, inlinepopups, insertdatetime, preview, media, searchreplace, print, contextmenu, paste, directionality, fullscreen, noneditable, visualchars, nonbreaking, xhtmlxtras, template, advlist`
extension.tinymce.skin	***String***	*Set the tinyMCE skin* *Values:* • *o2k7* `o2k7`
		Set the tinyMCE skin variant *Values:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

extension.tinymce.skin.variant	***String***	• *silver* `silver`
extension.tinymce.theme	***String***	*Set the tinyMCE theme* *Values:* • *simple* • *advanced* `advanced`
extension.tinymce.theme.buttons1	***String***	*Set elements present in first toolbar.* *Values:* • *okm_checkin* • *okm_cancelcheckout* • *okm_searchDocument* • *okm_searchFolder* • *okm_searchImage* • *newdocument* • *codehighlighting* • *bold* • *italic* • *underline* • *strikethrough* • *justifyleft* • *justifycenter* • *justifyright* • *justifyfull* • *styleselect* • *formatselect* • *fontselect* • *fontsizeselect* `okm_checkin, okm_cancelcheckout,`

		`okm_searchDocument, okm_searchFolder, okm_searchImage, newdocument, codehighlighting, |, bold, italic, underline, strikethrough, |, justifyleft, justifycenter, justifyright, justifyfull, styleselect, formatselect, fontselect, fontsizeselect`
extension.tinymce.theme.buttons2	***String***	*Set elements present in second toolbar.* *Values:* • *cutcopy* • *paste* • *pastetext* • *pasteword* • *search* • *replace* • *bullist* • *numlist* • *outdent* • *indent* • *blockquote* • *undo* • *redo* • *link* • *unlink* • *anchor* • *image* • *cleanup* • *help* • *code* • *insertdate* • *inserttime* • *preview* • *forecolor* • *backcolor*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`cut, copy, paste, pastetext, pasteword, |, search, replace, |, bullist, numlist, |, outdent, indent, blockquote, |, undo, redo, |, link, unlink, anchor, image, cleanup, help, code, |, insertdate, inserttime, preview, |, forecolor, backcolor`
extension.tinymce.theme.buttons3	***String***	*Set elements present in third toolbar.* *Values:* • *tablecontrols* • *hr* • *removeformat* • *visualaid* • *sub* • *sup* • *charmap* • *emotions* • *iespell* • *media* • *advhr* • *print* • *ltr* • *rtl* `tablecontrols, |, hr, removeformat, visualaid, |, sub, sup, |, charmap, emotions, iespell, media, advhr, |, print, |, ltr, rtl`
		Set elements present in fourth toolbar. *Values:* • *insertlayer* • *moveforward* • *movebackward* • *absolute*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

extension.tinymce.theme.buttons4	***String***	• *styleprops* • *cite* • *abbr* • *acronym* • *del* • *ins* • *attribs* • *visualchars* • *nonbreaking* • *template* • *pagebreak* `insertlayer, moveforward, movebackward, absolute, |, styleprops, |, cite, abbr, acronym, del, ins, attribs, |, visualchars, nonbreaking, template, pagebreak`

DXF applet viewer

Application comes with DXF applet viewer. If you observe problems on previewer text, you should configure the URL to indicate an extra fonts repository used in AutoCAD files. Simply copy the files to some public URL that can be resolved by desktop computer.

The font file names should be in lower case (rename if you have upper-case font file names).

Field / Property	*Type*	*Description*
extension.dxf.applet.debug.window	***Boolean***	*Enable debug in java console view. By default is disabled.* `false`
extension.dxf.applet.default.bigfont	***String***	*Font might be used by big text. By default is empty.* `arial`
extension.dxf.applet.default.font	***String***	*Set the default font. By default is empty.* `arial`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

extension.dxf.applet.fontdir	***String***	*Public URL with extra font repository. By default is empty.* `http://fonts.openkm.com`
extension.cadviewer.applet.debug	***Boolean***	*When is enabled the CADViewer applets shows extra debug information.*

Extratab extension

The ExtraTab allows to easily add a new tab with a name and specific URL.

Field / Property	*Type*	*Description*
extension.extra.tab.workspace	***List***	*Title and url pair. You can add single line (where the character ; is the separator between title and url) or several lines.* `OpenKM;http://www.openkm.com` *If **URL does not start with httextension.extra.tab.workspacep** it's considered as **OpenKM relative path** and will be merged with your OpenKM server URL like **YOUR_OPENKM_URL/url***

External preview extension

The External Preview feature helps you to preview documents based on external URL request.

Field / Property	*Type*	*Description*

extension.external.preview	*List*	*Each row couple of values is a preview content declaration, the format is:*
		mime-type;url
		`application/illustrator;http://localhost:8180/OpenKM/test`
		`application/epub+zip;http://localhost:8180/OpenKM/epub_pr`
		`application/x-mspublisher;http://localhost:8180/OpenKM/mspreviewer_prev`

External tab document extension

The External tab document feature shows extra tabs in tab document view based on external URL request.

Field / Property	*Type*	*Description*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

extension.external.tab.document	***List***	*The format for each line is **title;URL** where character ";" is used as separator between title of tab and URL of contents.* `OpenKM;http;//www.openkm.com`

External tab folder extension

The External Tab Folder feature helps you to show the tab content in properties tab based on external URL request when a node folder is selected.

Field / Property	*Type*	*Description*

extension.external.tab.folder	***List***	*Each row is a couple of values with a tab content declaration, the format is: **name;url***
		`name1;http://localhost:8180/OpenKM/test.jsp`
		`name2;http://www.openkm.com`

External tab mail extension

The External Tab Mail feature helps you to show tab content in properties tab based on external URL request when a node mail is selected.

Field / Property	*Type*	*Description*

extension.external.tab.mail	***List***	*Each row is a couple of values with a tab content declaration, the format is: **name;url***
		`name1;http://localhost:8180/OpenKM/test.jsp`
		`name2;http://www.openkm.com`

External tab record extension

The External Tab Record feature helps you to show tab content in properties tab based on external URL request when a node record is selected.

Field / Property	*Type*	*Description*
extension.external.tab.record	***List***	*Each row is a couple of values with a tab content declaration, the format is: **name;url*** `name1;http://localhost:8180/OpenKM/test.jsp`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`name2;http://www.openkm.com`

Menu options extension

The Menu options feature helps you to set new toolbar menu options. Each menu option opens popup with external URL.

Field / Property	*Type*	*Description*
extension.toolbar.menu	***String***	*Each row is a couple of values with a tab content declaration, the format is:* ***name;url;width;height*** `menu1; http://www.openkm.com;500;500` `menu2; http://www.openkm.com;700;700` `menu3; http://www.wiki.openkm.com;900;300` `menu4; http://www.forum.openkm.com;500;900` `menu5; http://docs.openkm.com;500;500` `menu6; http://issues.openkm.com;700;700`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Performance configuration parameters

Text extractor queue

There are several configuration parameters to adapt the document text extraction to your needs. Text extraction is one of the processes that can use a lot of hardware resources (memory and CPU's). Aggressive text extraction policy can decrease the performance of the application and affect the end user feeling with user interface.

When documents are uploaded are automatically added into ***Text extraction queue****. Can see Text extraction queue at* ***Administration > Statistics > Text extraction queue****.*

*There's a crontab task named "****Text extractor worker****" at* ***Administration > Cron tab*** *what control the indexing cycle*

> *When a document is uploaded, it will not be able to search by content until is processed by Text extraction queue.*

> *In case the application need indexing a lot of files per day, or you imported a considerable amount of files, consider create two crontab task to change these parameters between day (less aggressive) and night (more aggressive).*

> *Some mime-types like pdf or images (what needs OCR engine) can use a lot of CPU (top 100%).*

Field / Property	*Type*	*Description*
text.extraction.batch	***Boolean***	*Indicate the number of documents to be processed on each indexing cycle.* *When parameter "**text.extraction.concurrent" is enabled**, is a good practice to be multiple of "**text.extraction.threads**" value.* `20`
text.extraction.concurrent	***Boolean***	*Enable or disable concurrent text extraction concurrent threads feature.* `True`
text.extraction.threads	***Integer***	*Number of concurrent threads will be used for text extraction feature.* *This number should be less or equal the number of hardware cores.* `2`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*There's a crontab task named "**Mailextractor worker**" at **Administration > Cron tab** what control the indexing cycle*

> *When a mail is uploaded, it will not be able to search by content until is processed by Mail extraction queue. Furthermore, attachment won't appear until it is processed.*

Field / Property	Type	Description
mail.extraction.batch	**Boolean**	*Indicate the number of mails to be processed on each indexing cycle.* *When parameter "**mail.extraction.concurrent" is enabled**, is a good practice to be multiple of "**mail.extraction.threads**" value.* `20`
mail.extraction.concurrent	**Boolean**	*Enable or disable concurrent mail extraction concurrent threads feature.* `True`
mail.extraction.threads	**Integer**	*Number of concurrent threads will be used for mail extraction feature.* *This number should be less or equal the number of hardware cores.* `2`

OpenOffice LibreOffice conversion service

As part of application startup, OpenKM executes an OpenOffice or LibreOffice service. That's used internally for conversion purposes, for example to converting doc files to pdf.

OpenOffice or LibreOffice service can use a lot of hardware resources (CPU top 100%), that can decrease the performance of the application. A good practice is move OpenOffice or LibreOffice conversion service to another server.

Field / Property	Type	Description
system.openoffice.server	**String**	*URL to OpenOffice or LibreOffice service.* `http://192.168.1.34:8080/converter/convert`
system.openoffice.tasks	**String**	*Restart service after x conversions to prevent memory leaks.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		`5`

For more information read:

- *OpenOffice LibreOffice configuration parameters*
- *OpenOffice and LibreOffice configuration*

Download files and folders as ZIP file

It is not a good practice to download as ZIP files high volume of files. In most cases take in consideration the application export tool at ***Administration > Utilities > Repository export***.

Downloading files and folders as ZIP file can take a lot of time and hardware resources. First, the application will prepare a temporal folder in the server that will be used to create the ZIP file, finally will be forwarded to user interface. To prevent blocking user interface waiting for process completion there are two parameters that will notice the user has exceeded some limits (number of files or size) and the process will go into background mode.

Field / Property	*Type*	*Description*
download.zip.live.max.files	***Integer***	*Set the maximum number of files allowed to download in live mode.* `100`
download.zip.live.max.file.size	***String***	*Set the maximum size of all files allowed to download in live mode.*

For more information about these parameter read *User interface configuration parameters*.

Antivirus checker

There are several configuration parameters to adapt the antivirus checker to your needs. Antivirus checker is one of the process that can use a lot of hardware resources (memory and CPU's). Aggressive antivirus checker policy can decrease the performance of the application and affect the end user feeling with user interface. The Antivirus checker can work in live mode (as uploading process antivirus check the document and if a virus is detected a warning error is immediately raised and the document is not incorporated to the repository) or check in background mode.

Antivirus checker in live mode can dramatically decrease the user interface and the end user feeling. Take in consideration that to analyse some documents the antivirus can check it in 3 seconds or more that will be added to the uploading time process.

In case the application needs to analyse a lot of files per day, or you imported a considerable amount of files, consider create two crontab task to change these parameters between day (less aggressive) and night (more aggressive).

Field / Property	*Type*	*Description*
system.antivir	***String***	*Set the antivirus path.*
background.antivirus.check	***Boolean***	*Enable or disable background antivirus checker. By default is set to **false**.* *There's a **crontab** task (**Administration > Crontab**)named **Antivirus Checker Worker** that take control of background antivirus checker.* *When the background check is disabled the antivirus is processed while uploading any document. That can add extra time to the uploading process (in some cases more than 3 seconds is the time needed to antivirus for checking a document). When you do a huge import of files the total time to complete the operation can be multiplied by x3 or more with this option disabled, because as part. Our suggestion is set it enabled.* `true`
background.antivirus.check.threads	***Integer***	*Number of concurrent threads will be used for antivirus.* *This number should be less or equal the number of hardware cores.* `2`
background.antivirus.check.batch	***Integer***	*Indicate the number of documents be processed on each indexing cycle.* *When parameter "**background.antivirus.check" is enabled**, is a good practice to be multiple of "**background.antivirus.check.threads**" value.* `20`

For more information read Antivirus configuration parameters.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Re-indexing the whole Lucene repository

*Application can use two different ways of rebuilding Lucene indexes: sequential or parallel. By default sequential re-indexing is enabled, but you can select the mode using the "**hibernate.indexer.mass.indexer**" configuration parameter.*

Field / Property	*Type*	*Description*
hibernate.indexer.mass.indexer	***Boolean***	*When is enabled the rebuild Lucene indexes process goes into "**Parallel mode**".*

Mode	*Description*
Sequential mode	*You can use the "**hibernate.indexer.batch.size.load.objects**" configuration parameter to indicate to Hibernate how many object should handle every time. To avoid **OutOfMemory** problems, the repository needs to be re-indexed in batch. If the value of this property is too low, the performance will be bad but if is too high you can have **OutOfMemory** problems.* • ***hibernate.indexer.batch.size.load.objects**: batch size used to load the root entities.* `30`

Parallel mode	*You have several configuration properties to tune its performance:*
	• ***hibernate.indexer.batch.size.load.objects**: batch size used to load the root entities.*
	`30`
	• ***hibernate.indexer.threads.subsequent.fetching**: number of threads used to load the lazy collections related to the indexed entities.*
	`8`
	• ***hibernate.indexer.threads.load.objects**: number of threads used to load the root entities.*
	`4`
	• ***hibernate.indexer.threads.index.writer**: number of threads used to analyse the documents and write to the index.*
	`3`

Execution timeout

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*The Application executes external applications to process documents, for example extracts the text with an OCR engine, analyse with an Antivirus software, transforms documents to other formats among other actions. Sometimes these processes can take a lot of time or for some reason is not finished correctly and the process keep on the OS consuming resources. To prevent it, the parameter "**system.execution.timeout**" set the maximum allowed time of execution on external applications.*

Field / Property	Type	Description
system.execution.timeout	**Integer**	*Set the maximum allowed time of execution on external applications.* *The units are minutes. By default the configuration is set to 5 minutes.* `5`

Upload bandwidth

Sometimes you want to restrict the total bandwidth used by each user while uploading files.

Field / Property	Type	Description
upload.throttle.filter	**Boolean**	*Limit the total uploading bandwidth to 10kb/sec per user.*

Remote conversion service

On huge repository with a lot of concurrent users is a good idea to configure a server only for document transformation.

> *To enable this feature is needed installing a specific OpenKM service application. Contact with OpenKM technical staff for more information.*

Field / Property	Type	Description
remote.conversion.server	**String**	*Set the URL of the conversion service.* `http://192.168.2.1:8080/converter/convert`

Activity log actions

OpenKM can log a lot of information related to the activity of the users , but sometimes these actions don't need to be logged and fill your activity log table.

> *The table where activity log is stored - OKM_ACTIVITY - may grow quickly storing millions of records.*

*There is a configuration property name "**activity.log.actions**" where you can set which actions to log. By default this is set to the*

most common or interesting actions. You can use regular expressions to define these actions. Read Java Regex Tutorial for more info about Java regular expressions.

Field / Property	Type	Description

activity.log.actions	**List**	`LOGIN`
		`LOGOUT`
		`CREATE_.*`
		`DELETE_.*`
		`PURGE_.*`
		`MOVE_.*`
		`COPY_.*`
		`CHECKOUT_DOCUMENT`
		`CHECKIN_DOCUMENT`
		`GET_DOCUMENT_CONTENT.*`
		`MISC_TEXT_EXTRACTION_FAILURE`

For a complete list of action see Activity log.

Dashboard activity

OpenKM dashboard displays a global view of application activity. For it, application needs to process a huge number of information what can decrease the overal perfomance of the application (for example for getting the last documents uploaded into the system OpenKM might execute database slow queries). For it, we suggest on huge repositories disable dashboard activity.

Field / Property	Type	Description
dashboard.user.activity	**Boolean**	*Set to false for disabling the feature.*

Repository stats

This info is available at Administration > Stats and show some graphs with information about repository size and number of documents, folder, mails and records by context. In case of huge repositories these queries may take too much time to complete, so it's recommended to disable them:

Field / Property	Type	Description
repository.stats.enabled	**Boolean**	*By default is true. Set to false to disable repository stats.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Lucene

Lucene search engine can be disabled or configured in serveral ways. Depending on it, the behavior of the application and the performance over all might change.

*These configuration parameteres going into the "**openkm.properties**" configuration file.*

Field / Property	*Type*	*Description*
spring.jpa.properties.hibernate.search.autoregister_listeners	***String***	*Enable or disable Lucene search engine.* *Allowed values:* • *on* • *off* `on`
spring.jpa.properties.hibernate.search.default.worker.execution	***String***	*Set the Lucene worker execution mode. By default is set to "sync" which means that every entity modification won't end until the Lucene index has been properly updated. In case of "async" these entities modification will be faster because the Lucene part is added to a queue and processed in a background thread.* *Allowed values:* • *sync* • *async* `sync`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OpenOffice & LibreOffice configuration parameters

As part of the application startup, OpenKM executes an OpenOffice or LibreOffice service. That's used internally for conversion purposes, for example to converting doc files to pdf.

> *The OpenOffice or LibreOffice process started before OpenKM startup, can cause the failure of the service that OpenKM internally executes during startup.*

> *The OpenOffice or LibreOffice service can use a lot of hardware resources (CPU top 100%).*

Field / Property	*Type*	*Description*
system.openoffice.path	***String***	*Path to OpenOffice or LibreOffice.*
system.openoffice.dictionary	***String***	*Dictionary used by search engine tokenizer.*
system.openoffice.server	***String***	*URL to OpenOffice or LibreOffice service.* *This parameter replaces the use of system.openoffice.path parameter.* `http://192.168.1.34:8080/converter/convert`
system.openoffice.tasks	***String***	*Restart service after x conversions to prevent memory leaks.* `5`
system.openoffice.port	***String***	*Default port is 2002.* `2002`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

WebDAV configuration parameters

Field / Property	*Type*	*Description*
system.webdav.server	***Boolean***	*Enable WebDAV service. By default is disabled.* `false`
system.webdav.fix	***Boolean***	*Character ":" can cause problems with some WebDAV client with this parameter enabled replaces in all URL the character ":" to "_". By default is disabled.* `false`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CSV configuration parameters

The CSV ("Comma Separated Value") file format is often used to exchange data between disparate applications. The file format, as it is used in Microsoft Excel, has become a pseudo standard throughout the industry, even among non-Microsoft platforms.

Some application features import or export data in CSV (Comma Separated Value) like results in search view users in administration among others. The format of the output CSV file can be configured with several parameters to your needs.

Field / Property	*Type*	*Description*
csv.format.delimiter	***String***	*Set the delimiter used between columns. By default is used character "**;**".*
csv.format.ignore.empty.lines	***Boolean***	*Set if empty lines must be processed or not. By default the value is **false**.*
csv.format.quote.character	***String***	*Set the quote character used to enclose column values. By default is set to " character.*
csv.format.skip_header	***Boolean***	*Set if first row contains column names and must not be processed as values or not. By default is set to **true**.*
csv.format.comment.indicator	***String***	*Set character used to indicate the beginning of a comment. By default is set to #.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Antivirus configuration parameters

Antivirus is one of the processes that can use a lot of hardware resources (memory and CPU's). An aggressive antivirus policy can decrease the performance of the application and affect end users feeling with user interface.

Antivirus analyzer can work in two mode (depending the value of ***background.antivirus.check*** *parameter):*

- *Live mode.*
- *Batch mode.*

> *Antivirus checker in live mode can dramatically decrease the user interface end user feeling. Take in consideration that to analyse some documents the antivirus can take up to 3 seconds or more what will be added to the uploading time process.*

Field / Property	*Type*	*Description*
system.antivir	***String***	*Set the antivirus path.* `/usr/bin/clamscan`
background.antivirus.check	***Boolean***	*Enable or disable background antivirus checker. By default is set to* ***false****.* *There's a* ***crontab*** *task (* ***Administration > Crontab*** *) named* ***Antivirus Checker Worker*** *that take control of background antivirus checker.* *For more information Built in crontab tasks.* *When background check is disabled the antivirus is processed while uploading any document. That can add extra time to the uploading process be complete (in some cases more than 3 seconds is the time needed for the antivirus to check a document). For example when you do a huge import of files the total time to complete the operation can be multiplied by x 3 or more with this option disabled, Our suggestion is set it enabled.*
background.antivirus.check.threads	***Integer***	*Number of concurrent threads will be used for antivirus.* *This number should be less or equal the number of*

		hardware cores.
background.antivirus.check.batch	***Integer***	*Indicate the number of documents be processed on each indexing cycle.* *When parameter "**background.antivirus.check" is enabled**, is a good practice to be multiple of "**background.antivirus.check.threads**" value.*

Other configuration parameters

Property	Type	Description
hibernate.search.analyzer	**String**	*This parameter is used to change the default Apache Lucene Search E* `org.apache.lucene.analysis.standard.StandardAn` *More information at Configuring Lucene Analyzer.* *This parameter must be set on $TOMCAT_HOME/openk*
hibernate.search	**Boolean**	*This parameter can be used to disable Hibernate Search completely. T able to search document or folders.* *This parameter must be set on $TOMCAT_HOME/openk*
system.ocr.rotate	**String**	*This parameter forces OCR to process on the document rotated.* *The feature is useful for example in upside down scanned standard OCR direction will fails.* *The parameter is a collection of degrees separated by character ";"* `90;180;270;` *More information at Configuring OCR engine.*
system.ocr.spell.check	**Boolean**	*When is enabled the text extracted by OCR is validated by spell check* *The application must have an enabled dictionary, see the "**system.openoffice.dictionary**" at OpenOffice LibreOffic parameters.*
system.document.name.mismatch.check	**Boolean**	*This parameter is used to enable or disable the name check when a ne document is uploaded.*
max.search.results	**Integer**	*This parameter sets the limits of the search results. For example a val*

		never get more than 500 results of any query. `500` *The reason why the number of results is limited is that so retrieve a huge number of results, specially when the que api methods.*
system.demo	***Boolean***	*This parameter is used to switch openkm to demo mode.* *On production environment should never be enabled.*
system.keyword.lowercase	***Boolean***	*This parameter is used to force all keyword to lowercase.* *The application is case sensitive, that means a user can s "keyone" and "KeyOne" and for the application these two distinct. To prevent this problem is always good idea forc lowercase.*

tomcat.connector.uri.encoding	***String***	*By default Tomcat Connector encoding is set to "**ISO-8859-1**" but you this parameter.*
		*When you set the "**URIEncoding**" parameter in Tomcat t need to set this parmeter to "**UTF-8**".*
		`URIEncoding="UTF-8"`
		`ISO-8859-1`

mail.session.debug	***Boolean***	*Enable mail session debug. It's useful to detect problems when applic mails.*
mobile.theme	***String***	*There're five mobile themes ("a", "b", "c", "d", "e").* *More information about themes at jQuery mobile themes* `b`

plugin.debug	***Boolean***	*When enabled show debug information about application plugin featu*
principal.hide.connection.roles	***Boolean***	*Hide "**ROLE_USER**" and "**ROLE_ADMIN**" roles from available ro* *By default the hidden roles are set by "**default.user.role**" "**default.admin.role**" parameters value. Usually "**ROLE_** "**ROLE_ADMIN**".*
principal.database.filter.inactive.users	***Boolean***	*Hide inactive users for available users list.*
restrict.file.name	***String***	*Restrict file name extension that are not allowed.* *Each file name extension is separated by character ";".* `*~;*.bak`
restrict.file.mime	***Boolean***	*Restrict file name extension based on existing ones at **Administration***
restrict.duplicated.documents	***Boolean***	*Is not allowed to upload more than one time the same document.* *Internally the verification is based on the document chec*
native.sql.optimizations	***Boolean***	*For best performance, some database operations are done with SQL*
repository.stats.optimization	***Boolean***	*When is enabled the **Administration** > Statistics results need less time calculated.*
repository.content.checksum	***Boolean***	*When is enabled is calculated the document checksum for each uploa*
system.pdf.force.ocr	***Boolean***	*Force to processing all PDF files with the OCR engine.*
database.pagination.optimization	***Boolean***	*When enabled execute pagination optimization to increase the speed*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

store.raw.mail	***Boolean***	*When is enabled also it stores a the imported mails raw.*
store.node.path	***Boolean***	*When enabled is stored in database the path of each node.* *For performance reason on move, delete and rename ope do not enable this parameter. We only recommend to ena with less 100K nodes.*
stack.metadata.hidden.properties	***List***	*Filter metadata fields at metadata stack view on frontend UI.*
show.records.in.tree	***Boolean***	*When is enabled show record nodes at tree stack view on frontend UI*
shCoreEclipse.css	***String***	*CSS style used by syntax highlighter.* *More information at http://alexgorbatchev.com/SyntaxHi*
shThemeEclipse.css	***String***	*CSS style used by syntax highlighter.* *More information at http://alexgorbatchev.com/SyntaxHi*
system.catdoc.xls2csv	***String***	*Native XLS to CSV converter tool. Used in* ***NativeMsExcelTextExtra*** *other Ubuntu based distros is included in the* ***catdoc*** *package.* `/usr/bin/xls2csv ${fileIn}`
system.embedded.preview	***Boolean***	*When is enabled - default - the pre visualization use a web PDF viewe*
system.maintenance	***Boolean***	*When is enabled the application goes to "maintence mode", that mear okmAdmin user will be able to login.*
system.strings	***String***	*Configure Strings tool.* `/usr/bin/strings ${fileIn}`
system.zbar	***String***	*Configure bar code extractor tool.*

		`/usr/bin/zbarimg`
system.profiling	***Boolean***	*The Profiling stats feature gives low level value information to the ad* *application performance problems.* *You also can enable profiling from **Administration > Uti** [stats]().
system.ftp.port	***Integer***	*FTP port.* *On linux only root user is able to binding ports like 21, th is good idea binding ftp at another port.*
system.ftp.server	***Boolean***	*When is enabled the application enables the FTP server feature.*
user.keywords.cache	***Boolean***	*When enabled the keyword of the nodes are cached.* *The feature provide best perfomance with less database a*
user.item.cache	***Boolean***	*When enabled the application cached the number of nodes owned by u* *The feature provide best perfomances with less database calculating the nodes owned by each user.*
max.search.clauses	***Integer***	*Set the maximum number of allowed clauses in a search.*
max.search.results	***Integer***	*Set the limit of the results returned by a search to the repository.*
search.suggestion.minimal.results	***Integer***	*The minimal search results to provide suggestions. By default is 5.*
search.suggestion.returned.words	***Integer***	*The number of suggested words. By default is 3.*
checkin.comment.mandatory	***Boolean***	*Force to set comment when uploading new document (create or upda*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

html.syntaxhighlighter.core	***String***	*Set the syntax highlight css core.* *More information at SyntaxHighlighter.*
html.syntaxhighlighter.theme	***String***	*Set the syntax highlight css theme.* *More information at SyntaxHighlighter.*
kea.thesaurus.skos.file	***String***	*The SKOS file.* `/vocabulary/ag_skos_20070219.rdf` *When enabled restricts the number of keywords available nodes. The SKOS file must be based on the ontology file set at **kea.thesaurus.owl.file** property (Thesaurus config* *More information about SKOS (Simple Knowledge Orga*
kea.thesaurus.vocabulary.serql	***String***	*Query to retrieve SKOS terms.* `SELECT X,UID FROM {X} skos:prefLabel {UID} WHE ="en" USING NAMESPACE rdf=<http://www.w3.org/ syntax-ns#>, skos= <http://www.w3.org/2004/02/skos/core#>,rdfs= <http://www.w3.org/2000/01/rdf-schema#>,dc= <http://purl.org/dc/elements/1.1/>, dcterms= <http://purl.org/dc/terms/>, foaf= <http://xmlns.com/foaf/0.1/>`
stack.metadata.adapter	***String***	*Set the metadata stack implementation.* `com.openkm.plugin.stack.metadata.GenericStackI` *More information at Creating your own Stack Metadata A*
system.diffpdf	***String***	*Configure DiffPDF tool.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Windows

You need the command line version from https://www.qtrac.eu/diffpd

```
C:\openkm\extra\diffpdf\diffpdf.exe -qr ${file
${fileIn1} ${fileIn2}
```

Linux

```
/home/openkm/tomcat-8.5.34/bin/diffpdf ${file
${fileOut}
```

In case of you don't have an X Server installed on Linux, you would n

Xvfb, and the poppler-qt dependency. In Debian / Ubuntu both them c

way:

```
$ sudo apt-get install xvfb libpoppler-qt4
```

In Ubuntu 18.04:

```
$ sudo apt-get install libqtgui4
```

And add $TOMCAT_HOME/lib/diffpdf libraries to LD_LIBRARY_PA

In CentOS:

```
$ yum install xorg-x11-server-Xvfb poppler-qt
$ ln -s /lib64/libpoppler-qt4.so.14.4.0 /lib64/libpoppler-qt4.so.4
```

And uncomment the corresponding lines from the Tomcat service scri

/etc/init.d/openkm:

```
$ECHO "Starting Xvfb"
Xvfb :1 &
export DISPLAY=:1
```

and

```
$ECHO "Stopping Xvfb"
killall Xvfb
```

Sometimes you need to add the "export DISPLAY" to the ***$TOMCAT_HOME/bin/setenv.sh*** *file. For example in C needed. If the problem persist, try to modify the* ***system.di*** *"/usr/bin/xvfb-run" command, like this:*

		/usr/bin/xvfb-run /home/openkm/tomcat-8.5.34/bin/diffp ${fileIn2} ${fileOut}
version.append.download	***Boolean***	*Append the document version in the document file name when downlo from OpenKM.*
cmis.dateformat	***String***	*Pattern to transform date and time values from CMIS format to Open.* *More information at Java 8 SimpleDateFormat* *By default the value is: **yyyyMMddHHmmss***
webservices.visible.properties	***List***	*List of configuration properties what are granted to view by users wit grants from webservices.*
rss.news	***Boolean***	*Enable or disable new in the login page.*
rss.news.box.width	***Integer***	*Width of the news box in the login page.* `300`
rss.news.max.size	***Integer***	*Maximum number of news retrieved from the news server.* `10` *News are cached from OpenKM server and refreshed one means any change on this parameter will take effect tomo OpenKM service.*
rss.news.visible	***Integer***	*Number of visible news at the same time.* `1`
trash.retention.days	***Integer***	*It indicates the maximum number of days nodes in the trash will be av purge will be executed. Default value is **0 indicates unlimited**.* *There's a Crontab task named "Purge Old Trash Nodes"*

		`30`
update.info	***Boolean***	*Application daily connects to OpenKM server in Internet to get news.* `true`
file.plan	***Boolean***	*Enable or disable the file plan feature.* `false`
decimal.pattern	***String***	*Set the decimal pattern.* `.`
browser.natural.sort	***Boolean***	*Enable or disable natural sort.* *Natural sort is done at database engine level.* *For MySQL and MariaDB download and install **natural** database.* *For another database engine, you need to create your ow "**okmNaturalSort(String, Integer, String)**".*
min.free.disk.percent	***Integer***	*This is the minimum percentage used to alert about low free disk spac* `20`
sanitize.lucene.fields	***Boolean***	*When enabled sanitize metadata fields.* *Because OpenKM metadata groups use characters reserv ":", when they are indexed by Lucene, these fields must b used in Working with Lucene queries.* *After enabling this parameter you must reindex the whole look at Rebuild indexes.*
search.by.attachment.parent	***Boolean***	*Enable or disable the search of documents in emails (attachments).* `false`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Experimental configuration parameters

Property	*Type*	*Description*
unit.testing.folder	***String***	*Folder used by testing process.* `/okm:root/okmTesting`
unit.testing.user	***String***	*User used for unit testing.* `okmAdmin`
unit.testing.password	***String***	*Password of the user used for unit testing.* `admin`
unit.testing.url.base	***String***	*The unit test URL.* `http://localhost:8080/OpenKM/`
extension.digicert.base.path	***String***	*OpenKM base folder:* `/okm:root/invoices`
extension.digicert.keystore.file	***String***	*Path of the keystore file.* `/home/openkm/certificate.p12`
extension.digicert.keystore.password	***String***	*The password of the keystore.*
extension.digicert.keystore.type	***String***	*The keystore type (JKS or PKCS12)* `JKS`
extension.digicert.private.key.alias	***String***	*The alias of the certificate.* `b138bef79c0104378101b77b71922c61_9463f621-75bc-4ff7-b832-08ea82e57a16`
extension.digicert.signature.algoritm	***String***	*The algorithm of the certificate.* `rsa`

extension.digicert.type	***String***	*Values can be "**month**" or "**quarterly**".* `month`
extension.digicert.upload.base.path	***String***	*OpenKM path where the documents are upload.* `/okm:root/invoices/upload`

extension.digicert.validate.properties	***String***	*Metadata group fields.*
		`okp:digicert.num.recepcion`
		`okp:digicert.num.factura`
		`okp:digicert.fecha.factura`
		`okp:digicert.nif.emisor`
		`okp:digicert.iva.general`
		`okp:digicert.iva.reducido`
		`okp:digicert.iva.superreducido`
		`okp:digicert.base.general`
		`okp:digicert.base.reducido`
		`okp:digicert.base.superreducido`
		`okp:digicert.base.siniva`
		`okp:digicert.total`
		`okp:digicert.year`
		`okp:digicert.period`

amazon.access.key	***String***	*The Amazon s3 access key.* *Must be configured to export repository content to Amazon S3.*
amazon.secret.key	***String***	*The Amazon s3 secret key.* *Must be configured to export repository content to Amazon S3.*
rss.news	***Boolean***	*Enable or disable new in the login page.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

rss.news.box.width	***Integer***	*Width of the news box in the login page.* `300`
rss.news.max.size	***Integer***	*Maximum number of news retrieved from the news server.* `10` *News are cached from OpenKM server and refreshed one time by day. That means any change on this parameter will take effect tomorrow or restarting OpenKM service.*
rss.news.visible	***Integer***	*Number of visible news at the same time.* `1`
check.node.deletion.integrity	***Boolean***	*Enable or disable node deletion integrity check. For example, if the node is used in a workflow or any of their children can be deleted, in case of folders or records. By default is enabled, and you should only disable it in case of deletion of massive nodes.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutoCAD configuration parameters

These parameters are used in AutoCAD integration.

Property	*Type*	
extension.cadviewer.autoxchange	***String***	*Tailormade AutoXchange AX2017 tool. This tool is used to convert a given* `/home/openkm/tomcat-7.0.61/bin/ax2017`
extension.cadviewer.linklist	***String***	*Tailormade LinkList tool. This tool is used to extract text, annotations from* `/home/openkm/tomcat-7.0.61/bin/ll2014`
extension.cadviewer.dwglist	***String***	*Configure DWGList tool. This tool is used to extract XRefs from a given DW* `/home/openkm/tomcat-7.0.61/bin/dwglist`
extension.cadviewer.sibling.xrefs	***Boolean***	*When this parameter is set to* ***true*** *the AutoCAD files's references have to be keep on mind that you should avoid document name duplication. There is ar*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Thesaurus configuration

*A **thesaurus** is a book that lists **words grouped together according to similarity of meaning** (containing synonyms and sometimes antonyms), in contrast to a dictionary, which contains definitions and pronunciations.*

*In Information Science, Library Science, and **Information Technology, specialized thesauri are designed for information retrieval**. They are a type of controlled vocabulary, for indexing or tagging purposes. If you want more information about the term you can take a look at http://en.wikipedia.org/wiki/Thesaurus.*

There's a great utility http://protege.stanford.edu/ for creating and maintaining your own thesaurus. Specific thesauri can be found on the Internet. For example, Agrovoc (FAO), NASA Thesaurus, UNESCO Thesaurus, Public administration Thesaurus among others. Some of them are free and some have usage restrictions.

In order to correctly configure OpenKM to use a thesaurus the following configuration properties must be set:

Property	Description
kea.thesaurus.owl.file	*The ontology file (Web ontology file)* `vocabulary/file.owl`
kea.thesaurus.base.url	*The base URL.* `http://www.someweb.org`
kea.thesaurus.tree.root	*Query to get the first node.* `SELECT DISTINCT UID, TEXT FROM {UID} Y {OBJECT}, {UID} rdfs:label {TEXT} ; [rdfs:subClassOf {CLAZZ}] where not bound(CLAZZ) and lang(TEXT)="en" USING NAMESPACE foaf= <http://xmlns.com/foaf/0.1/>, dcterms= <http://purl.org/dc/terms/>, rdf=<http://www.w3.org/1999/02/22-rdf-syntax-ns#>, owl= <http://www.w3.org/2002/07/owl#>, rdfs= <http://www.w3.org/2000/01/rdf-schema#>, skos=<http://www.w3.org/2004/02/skos/core#>, dc= <http://purl.org/dc/elements/1.1/>`

kea.thesaurus.tree.children	*Query to get children node.* `SELECT DISTINCT UID, TEXT FROM {UID} rdfs:subClassOf {CLAZZ}, {UID} rdfs:label {TEXT} where xsd:string(CLAZZ) = "RDFparentID" and lang(TEXT)="en" USING NAMESPACE foaf=<http://xmlns.com/foaf/0.1/>, dcterms=<http://purl.org/dc/terms/>, rdf=<http://www.w3.org /1999/02/22-rdf-syntax-ns#>, owl=<http://www.w3.org/2002/07/owl#>, rdfs=<http://www.w3.org/2000/01/rdf-schema#>, skos=<http://www.w3.org/2004/02/skos/core#>, dc=<http://purl.org/dc/elements/1.1/>`

Take a look a full configuration example at Creating your own thesaurus.

Some interesting URL:

- *A Method for Converting Thesauri to RDF/OWL*
- *OWL to SKOS online conversor*
- *SKOS to OWL online conversor*
- *SKOS Simple Knowledge Organization System*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FTP configuration parameters

Field / Property	*Type*	*Description*
system.ftp.server	***Boolean***	*Enable FTP service. By default is disabled. You have to restart OpenKM after changing this parameter.* `false`
system.ftp.port	***Integer***	*The port where the FTP service is listening. By default is 2121 although the default FTP port is 21 because it's a reserved port and OpenKM should be executed as an administrator to be able of using this port. You have to restart OpenKM after changing this parameter.* `2121`
system.ftp.data.port.range	***String***	*Port range used in the server for data transmission. You should also open these ports in your firewall. By default the range is 60000 to 60500 and they have to be separated by a colon. You have to restart OpenKM after changing this parameter.* `60000:60500`

If you change any of these configuration parameters it won't take effect until OpenKM have been restarted.

CIFS's FTP server does not support Passive Mode. So, it won't work with file browsers such as Nautilus or Nemo.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Two factor authentication configuration parameters

In this section you can find the most common configuration parameters related to two factor authentication.

Field / Property	*Type*	*Description*
two.factor.authentication	***Boolean***	*Enable or disable two factor authentication functionality. False by default.*
two.factor.authentication.domain	***String***	*Domain name used to configure Google Authenticator. The defaul value is "openkm.com".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Multi-tenant configuration parameters

You can have some configuration parameters which are global for all the tenants, and some which are related to a given tenant. How can you do that? If you go to ***Administration > Configuration*** *you are able to see a option in the top right corner called "Tenant configuration". If you click in this option you will have access to these specific tenant configuration properties. By default it is empty, so you only can see an empty list.*

By clicking in the green plus icon you can create a new configuration property. For example, we are going to configure a different login image for each tenant. To do so follow these steps:

- *Go to the desired tenat (Menu -> tenant)*
- *Create a* ***File*** *property called* ***logo.login*** *and choose an image.*
- *Create a* ***String*** *property called* ***tenant.domain*** *and write the domain used by this tenant.*

To get this working, this tenant should be accessed by a given domain. For example, let's imagine you have a tenant called "Floppy" and you have the server configured to access the OpenKM instance using this domain floppy.okm-instance.com, so in this case the "tenant.domain" value should be "floppy.okm-instance.com". To create different alias you have to create the subdomain in your DNS server and make it to resolve to the real server IP. When OpenKM detects it is being accessed by this domain, it will show the configured ***logo.login*** *image.*

The same works for:

- ***logo.favicon***
- ***logo.mobile***
- ***logo.report***
- ***logo.tiny***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

File plan

A file plan is a comprehensive outline that includes the records series, file organization, active file locations, file transfer instructions, file retention and disposition instructions, and other specific instructions that provide guidance for effective management of records, including vital records.

File plans specify how records are to be organized once they have been created or received, provide a "roadmap" to the records created and maintained by an organizational unit, and facilitate dispositioning of the records.

Enable the File plan

Mandatory

Change the next OpenKM configuration parameters:

Field / Property	*Type*	*Description*
security.access.manager	***String***	*com.openkm.plugin.access.DbFilePlanAccessManagerWrapper*
file.plan	***Boolean***	***true***

After you updated the values, you must restart OpenKM service.

Optional

In the profiles we suggest the next changes:

- *In the "**Components tab**" disable the "**Security checkbox**" from all the tabs (Tab document, Tab Folder, Tab mail, Tab record).*
- *In the "**Components tab**" enable the "**Bussiness activity**" checkbox.*
- *In the "**Menu tab**" enable the "**File plan**" checkbox.*
- *In the "**Other tab**" enable the "**File plan**" checkbox.*

How to construct the File plan

After inventorying all of the records maintained by your office, you are ready to create your office file plan. A file plan describes every type of records in the organization, the location where they should be stored, the rules applying to them, the retention schedule and timeline, manner of their disposition, and the person or persons responsible for their management.
A file plan usually consists of:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Description of the kinds of documents identified to be records.*
- *The classification of the records into wide range of categories.*
- *Details about the location of the stored records.*
- *Description of the retention schedule and period.*
- *Making the distinction and assigning the responsibility for certain types of records.*

In the file plan every document must have:

Field	*Type*	*Sample*
Name	***String***	*Records common*
Code	***Number***	*101*
Definition	***Document type***	• *Document* • *Mail* • *Record* • *Electronic record (record)*
Section	***Section***	*100- Common section*
Property group	***Metadata***	*okg:technology* *okg:consulting*
Security access Manager	***Plugin***	*Default access manager*
Security by user	***Users***	*okmAdmin*
Security by roles	***Roles***	*ROL_USER*
Catalog manager	***Plugin***	*Business catalog manager*
Record manager	***Plugin***	*Default record manager*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Disposition manager	***Plugin***	*EmptyFilePlanDispositionManager*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

MIME types

The application supports several documents MIME type. Every MIME type has one or several file extensions which are used to match the uploaded document MIME type.

> *Based on the MIME type associated to the uploaded document, the application chooses the associated text extractor. Text extractor works only for a few MIME types that can easilly be extended.*
>
> *For example, if you upload a "sample.doc", it's detected MIME type is "application/msword" because its extension is ".doc". In this case, the text extractor class is "com.openkm.extractor.MsWordTextExtractor".*

The detected document MIME type is also used to display an icon which represents the type of document. You can change this icon if you like.

There is list of common documents MIME types already registered in OpenKM.

> *If you upload a document which MIME type is not in the list, it will be detected as "application/octet-stream".*

Add new MIME type

- *Click **new MIME Type** icon* .
- *In Create mime type view **fill the required fields**:*

Field	*Value*
Name	*The MIME type name* `application/epub+zip`
Description	*General purpose description.* `EPUB`
Extensions	*Associated document extensions.* *In case multiple extensions, each one should be separated by character ","* `epub`
	Image file.

Image	*Image size should be 16x16 pixels.*
Search	*When is checked it means it will be shown in the application user interface at search view MIME type list.*
Live Edit	*When is checked it means that this mime type will be edited using Live Edit when enabled.*

- *Click **Create button**.*

Edit MIME Type

- *Click **Edit icon** .*
- *In **Edit MIME type view** make changes in fields.*
- *Click **Edit button**.*

Delete MIME type

- *Click **Delete icon** .*
- *In **Delete MIME type view** click **Delete button**.*

Backup MIME types

- *Click at top right the **Export Mime type icon** .*
- *After a few seconds the application will download an SQL file.*

Restore MIME types

Restoring MIME types will remove actual values and will be replaced by the original ones.

- *Go to **bottom page**.*
- *Select **MIME type sql file**.*
- *Click **Import MIME types button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Statistics

In the statictics view, a graphical information about repository size and it's distribution on the distinct root nodes (taxonomy, personal, trash, etc.) is shown. Also is shown the basic server information like memory and disk usage.

To understand how your repository is growing and making hard disk estimations, these pie charts are interesting:

- *Documents by context.*
- *Documents size by context.*
- *Folders by context.*
- *Mails by context.*
- *Record by context.*

Some pie charts describe the distribution by context. These contexts are:

- *Taxonomy (the /okm:root context)*
- *Template (the /okm:template context, where you can place your document templates)*
- *Personal (the /okm:personal context, where each OpenKM user can stores their personal - not shared - documents)*
- *Mail (the /okm:mail context, where the user mails are stored by default)*
- *Trash (the /okm:trash context, which stores the deleted document, folders, etc.)*

If the trash context is too large the trash bin should be emptied. Once you are sure these documents and folders can be really deleted, you should empty your trash or individually purge the documents. Remember that each user has its own trash and every user is responsible of purging their own one. There's a feature ***Administration*** *>* ***Utilities*** *>* ***Purge trashes*** *that helps on it.*

Check the status of the Pending task queue

Pending tasks are only used in some application sections where an user action can takes longer than usual.

- *Click at top right the menu option* ***Pending task queue***.
- *In* ***Pending task queue view*** *is shown the status of the tasks. This is a list of the task to be processed or in progress.*

For example, if you want to change the permissions in a folder with lots of documents and / or a very depth structure this actions will be promoted to pending task. Meanwhile the user can do more thinks while this one is executed in background. If the number of document to process is huge, this task may take hours to complete.

Check the status of the Text extraction queue

When upload a document to the application, its content is not extracted inmediatelly but enqueued for this.

- *Click at top right the menu option* ***Pending text extractor queue***.

- *In **Pending text extractor queue view** is shown the status of the text extractor process. This is a list of the documents to be processed or in progress.*

More information at Performance configuration parameters.

Sometimes you might be interested on process all the documents again into the text extraction queue (you have introduced new text extractor, etc ...), you can find more information at Database description about how doing it.

Check the status of the Antivirus check queue

When upload a document to the application, the antivirus analysis can be done just in time or enqueued for this.

This options is only shown when the application has been configured to antivirus analyze on batch queue.

- *Click at top right the menu option **antivirus check queue**.*
- *In **Antivirus check queue view** is shown the status of the antivirus analyze process. This is a list of the documents to be processed or in progress.*

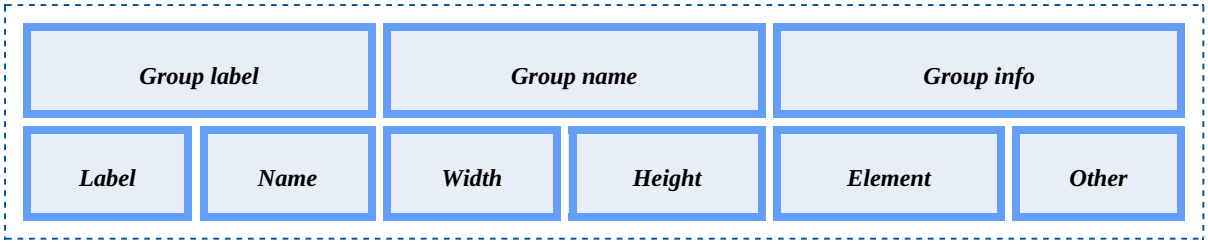
More information at Performance configuration parameters.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata

In the Metadata view are shown the registered metadata groups. In this view, it can create, modify or remove metadata definitions.

Property groups are shown in a table structure. First row (three columns) corresponds to the general Property group definition. The next rows are fields definition in Property group.



Description:

Property	*Description*
Group label	*Group label is shown in the user interface at Property group tab.*
Group name	*Unique group name identifier.* *Two property groups cannot have the same name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*
Group info	*Shows information about:* • *Visibility (property group is hidden from user interface).* • *Read only (property group can not be modified from user interface).*
Field label	*Field label is shown in the user interface in the Property group tab,*
Field name	*Unique property group field identifier.* *Two property group fields can't have the same name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Field width	*This parameter is used for rendering the width of the field in the user interface view.*
Field height	*This parameter is used for rendering the height of the field in the user interface view.*
Element	*This parameter sets the element type. It's used for rendering several values in the field in the user interface view.* *Values:* • *CheckBox* • *Input* • *Iframe* • *Select* • *Separator* • *SuggestBox* • *Text* • *TextArea*
Others	*Shows field extra information.*

Edit metadata

To add, modify or delete metadata there's only one operation for all, edit the metadata xml definition.

- *Click top right* ***Edit metadata***.
- *In Edit view,* ***modify the XML***.
- *At bottom right, click* ***Edit button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata XML definition

The application metada is based on a formal XML definition. The DTD (Document Type Definition) defines the structure and the legal elements and attributes of an XML document.

*You can see latest DTD in our website http://www.openkm.com/dtd/property-groups-**{major}**.**{minor}**.dtd where **{major}** and **{minor}** values are two numbers (major release and minor release). For example the location for version 3.7 of the DTD is http://www.openkm.com/dtd/property-groups-3.7.dtd*

Each application version supports only a couple of DTD definitions, read Application version compatibility table to see which are supported by your application.

Basic XML skeleton

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>

</property-groups>
```

- *The **DOCTYPE** is a formal definition of some DTD. In the previous example the **property-groups-3.7.dtd**.*
- *The XML begins with **<property-groups>** tag and ends with **</property-groups>**. Into this hicheracy are defined the metada groups.*

Metadata group definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">

  </property-group>
</property-groups>
```

- *Each **<property-group>...</property-group** block identify the **begining and end** of new **metadata group definition**.*
- *Each metadata group has a **label** - used by user interface - **Consulting**.*
- *Each metadata group has a **unique name**, **okg:consulting**.*
- *The metadata group name always must starts with "**okg:**" (**O**pen**KM** **G**roup).*
- *Must not use special characters for name.*

Attribute	*Description*	*Required*
		true

label	*The text shown as a label in the user interface.*	
name	*Unique field identifier.* *Two metadata groups can't have the same name.* ***Name must be unique****.* *The name must start with "**okg:**".* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
visible	*Show or hide the metadata.* *Allowed values are:* • *true.* • *false.* *By default the metadata are visible.*	*false*
readonly	*Metadata values can not be modified from user interface only from API.* *Allowed values are:* • *true.* • *false.* *By default the metadata are modificable.*	*false*
defaultValueClassName	*Set custom JAVA class for setting initial default values.*	*false*

Basic example

Available field types:

- *Checkbox*
- *Input*
- *Iframe*
- *Select*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Separator*
- *Suggestbox*
- *Text*
- *TextArea*

More information about field type elements at Metadata fields

*It's a good practice to use the metadata group name as the basis to create metadata field name. For example **okp:consulting.code** use **okg:consulting** as base (changing **okg** to **okp**). That helps to find out which is the metada group asociated to each metadata field.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Input label" name="okp:consulting.input1" />
    <separator label="Separator label" name="okp:consulting.separator" />
    <input label="Input label" name="okp:consulting.input2" />
  </property-group>
</property-groups>
```

Using file system DTD

If the server has no access to the Internet, then the DTD file must be accessible in some way, for example can store a copy in your application server.

The XML definition must be changed to refer to the new DTD location, otherwise you will get an application error while processing DTD file.

Can be a file system (example below) or a valid intranet URL.

```
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "file:///home/openkm/property-groups-3.7.dtd">
```

- *In the example, **/home/openkm/property-groups-3.7.dtd** is the file system path to the file.*

How metadata is stored into OpenKM

When you register a new metadata definition in OpenKM the metadata values are stored in two separated tables. One table for current metadata values and the other for history metadata values.

Sample:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
```

```
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Input label" name="okp:consulting.input1" />
    <separator label="Separator label" name="okp:consulting.separator" />
    <input label="Input label" name="okp:consulting.input2" />
  </property-group>
</property-groups>
```

After registering the previous metadata definition will be created two tables with these columns:

- ***OKM_PGRP_CUR_CONSULTING*** *(used for current metadata values)*
  - ***RGT_UUID****: ID of the node which has this metadata*
  - ***RGT_PRO_INPUT1****: Value of property okp:consulting.input1*
  - ***RGT_PRO_INPUT2****: Value of property okp:consulting.input2*
  - ***RGT_PRO_SEPARATOR****: Value of property okp:consulting.separator*
- ***OKM_PGRP_HIS_CONSULTING*** *(used for history metadata values)*
  - ***RGT_UUID****: ID of the node which has this metadata*
  - ***RGT_VERSION****: ID of the related version*
  - ***RGT_PRO_INPUT1****: Value of property okp:consulting.input1*
  - ***RGT_PRO_INPUT2****: Value of property okp:consulting.input2*
  - ***RGT_PRO_SEPARATOR****: Value of property okp:consulting.separator*

The prefix used for all the columns will be ***RGT_****.*

The column with name ***RGT_UUID*** *contains the value of the node UUID.*

Be careful with the Oracle table name and column name 32 characters limitation.

Default column size by type

Type	*Size*
Input	*128 characters*
Select	*128 characters*
SuggestBox	*128 characters*
TextArea	*256 characters*

In case you have metadata values larger should increase the value of the column with the attribute " ***dbColumnSize****". See the*

sample below:

*The value of the attribute "**dbColumnSize**" will only be applied the first time you register the metadata group or each time you add a new file. After the column has been created in the database, although you change the value "**dbColumnSize**" of the attribute, the column size will not be updated. In case you want to update the column size after creation, should apply changes from database client directly.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Input label" name="okp:consulting.input1" dbColumnSize="256"/>
    <separator label="Separator label" name="okp:consulting.separator" />
    <input label="Input label" name="okp:consulting.input2" dbColumnSize="512"/>
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata fields

These are the available field types:

Field type	*Description*	*Visible in the metadata stack panel*
Checkbox	*Metadata Checkbox field*	*Yes.*
Input	*Metadata Input field*	*No.*
Iframe	*Metadata Iframe field*	*No.*
Select	*Metadata Select field*	*Yes.*
Separator	*Metadata Separator field*	*No.*
Suggestbox	*Metadata Suggestbox field*	*Yes.*
Text	*Metadata Text field*	*No.*
TextArea	*Metadata TextArea field*	*No.*
VerticalPanel	*Metadata VerticalPanel field*	*No.*
HorizontalPanel	*Metadata HorizontalPanel field*	*No.*

Some field types can have children elements, validator or option elements. This hierarchy is permitted:

- *metadata group*
  - *Checkbox*
    - *Validator*
  - *Input*
    - *Validator*
  - *Iframe*
  - *Select*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Option*
- *Validator*

- *Separator*
- *Suggestbox*
  - *Validator*
- *Text*
- *TextArea*
  - *Validator*
- *VerticalPanel*
  - *CheckBox*
  - *Input*
  - *Iframe*
  - *Select*
  - *Separator*
  - *Suggestbox*
  - *Text*
  - *TextArea*
  - *VerticalPanel*
  - *HorizontalPanel*
- *HorizontalPanel*
  - *CheckBox*
  - *Input*
  - *Iframe*
  - *Select*
  - *Separator*
  - *Suggestbox*
  - *Text*
  - *TextArea*
  - *VerticalPanel*
  - *HorizontalPanel*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Checkbox field

Checkbox

Best used to represent boolean data:

Attribute	*Description*	*Required*
label	*The text shown as a label in the user interface.*	*true*
name	*Unique field identifier* *Two metadata can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use a metadata group name as basis for a metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.checkbox", changing "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
description	*General description.*	*false*
readonly	*Set field will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*
dbColumnSize	*Set the size of the database field where the value is stored.* *By default, fields are stored in a database column with 128 characters size. Because your data length might be bigger than 128 characters the administrator should update the default column size.* *Take a look at Metadata XML definition documentation section for a deep explanation about metadata storage in OpenKM database,*	
Child elements:		

- *Validator. For more information see Metadata Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <checkbox label="Checkbox label" name="okp:consulting.checkbox" />
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Iframe field

The application will render into an iframe object the selected web url. In the url will be passed as query parameters the uuid and propertyGroup.

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two metadata can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use a metadata group name as basis for the metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.iframe", changing "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
url	*Set the url of the web page to be load in the iframe.* *There're set two parameters to the request:* • *uuid (the node of the actual object selected in the user interface).* • *propertyGroup (the name of the metadata group).* *If your url value is http://www.openkm.com the url called into iframe will be http://www.openkm.com?uuid=03ced012-63cc-4feb-ac26-bb12ded86f73&propertyGroup=okg:consulting* *It's a good practice the URL starts with http:// or https://*	*true*
width	*The width of the HTML element. It is advisable to set the attribute width to 100% so that it is adjusted to the screen*	*false.*
height	*The height of the HTML element.*	*false.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <iframe label="Iframe label"  name="okp:consulting.iframe" url="http://www.openkm
  </property-group>
</property-groups>
```

Metadata Select field

Attribute	*Description*	*Required*
label	*The text shown as label in the user interface.*	*true*
name	*Unique field identifier.* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "****okp:****".* *It's a good practice to use metadata group name as basis for metadata field name. For example use "****okg:consulting****" has been used for field name "****okp:consulting****.select", changin "****okg****" to "****okp****". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
type	*Type value. Allowed types:* • *simple (single choice allowed).* • *multiple (multiple choices allowed).*	*true*
table	*Metadata table name.* *The application can store values into a metadata table, that can be used in this query.*	*
width	*The width of the HTML element.*	
height	*The height of the HTML element.*	
optionsQuery	*A metadata query used to get the selected options.*	*
className	*A class used to get the selected options.*	*
	When adding metadata to documents, the application can suggest automatically some values based on document content analysis. By default the application has some	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

suggestion	*implementations that can easilly be extended.* • ***com.openkm.form.suggestion.DocumentContentContainsSuggestion**: Gets suggestions based on document's content (use String.contains)* • ***com.openkm.form.suggestion.DocumentContentTokenizerSuggestion**: Gets suggestions based on document's content (Use StringTokenizer).* • ***com.openkm.form.suggestion.DocumentTermsSuggestion**: Gets suggestions based on document's term vectors.* *For more information read Creating your own Suggestion Analyzer.*	*false*
parentElement	*Parent field name value. In this case the value has a hierarchical relationship with other field values (the parent).*	*false*
description	*General description.*	*false*
readonly	*Set field will not be able to be modified in the user interface. By default the value is "**false**".*	*false*
dbColumnSize	*Set the size of the database field where the value is stored.* *By default, fields are stored in a database column with 128 characters size. Because your data length might be bigger than 128 characters the administrator should update the default column size.* *Take a look at Metadata XML definition documentation section for a deep explanation about metadata storage in OpenKM database,*	

Child elements:

- *Option. Couples of XML **<option label="label text" value="unique identifier value"/>** values or **<option label="label text" value="unique identifier value" parentValue="parent unique identifier value"/>** in case is set a hierarchical relationship.*
- *Validator. For more information see Metadata Validator element.*

***** There're tree options to get option values:*

- *Directly defined in XML.*
- *Metadata table (table + optionsQuery).*
- *className.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

One of these options must be defined, otherwise you will get an error. It can not done as a combination of several of them, only one can be choosen.

Basic select example

- *Type is equal to **simple**.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="select label" name="okp:consulting.select" type="simple">
      <option label="one" value="001" />
      <option label="two" value="002" />
      <option label="three" value="003" />
     </select>
  </property-group>
</property-groups>
```

Allow multiple choices

- *Type is equal to **multiple**.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="select label" name="okp:consulting.select" type="multiple">
      <option label="one" value="001" />
      <option label="two" value="002" />
      <option label="three" value="003" />
     </select>
  </property-group>
</property-groups>
```

Hierarchical relationship

- *The field option values of "**okp:consulting.children**" have parent relation with "**okp:consulting.parent**". It's defined by **parentElement="okp:consulting.parent"**.*
- *The option value "**benefits**" of field "**okp:consulting.children**" has a parent value "**directives**" from field "**okp:consulting.parent**".*

Tree relationships:

- *directives*
    - *benefits*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *elegibility*
- *services*

- *handbook*
  - *basic*
  - *normal*
  - *advanced*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="Parent" name="okp:consulting.parent" type="simple">
      <option value="directives" label="Directives" />
      <option value="handbook" label="Handbook Paragraphs" />
    </select>
    <select label="Children" name="okp:consulting.children" type="simple" parentEleme
      <option value="benefits" label="Benefits" parentValue="directives" />
      <option value="elegibility" label="Eligibility" parentValue="directives" />
      <option value="services" label="Services" parentValue="directives" />
      <option value="basic" label="Basic" parentValue="handbook" />
      <option value="normal" label="Advanced" parentValue="handbook" />
      <option value="advanced" label="Normal" parentValue="handbook" />
    </select>
  </property-group>
</property-groups>
```

Based on SQL query

We can also get these option values using a SQL query. This SQL sentence should be put on the ***optionsQuery*** *attribute, but the* ***table*** *attribute should be empty (otherwise this query is used as a metadata query). Note that the first element returned in every row is the option value and the second one is the option label.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Location" name="okg:location">
    <select label="Country" name="okp:location.country" type="simple"
            optionsQuery="select CT_ID, CT_NAME from COUNTRY order by CT_NAME" />
  </property-group>
</property-groups>
```

The table used in this sample is defined as:

```
create table COUNTRY (
  CT_ID varchar(2),
  CT_NAME varchar(32),
  primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Based on className

To help getting a list from external applications or other purposes, your own class can be built to retrieve the option values from those systems .

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="select label" name="okp:consulting.select" type="simple"
            className="com.openkm.plugin.form.values.OptionSelectUserList" />
  </property-group>
</property-groups>
```

For more information read Creating your own Option Select Values plugin.

Suggested options

Based on the content analyzer the application can suggest some options values. At the user interface the suggested options will be shown in green.

The application has some implementations that can easily be extended.

- ***com.openkm.form.suggestion.DocumentContentContainsSuggestion**: Get suggestions based on document content (use String.contains)*
- ***com.openkm.form.suggestion.DocumentContentTokenizerSuggestion**: Get suggestions based on document content (Use StringTokenizer).*
- ***com.openkm.form.suggestion.DocumentTermsSuggestion**: Get suggestions based on document term vectors.*

For more information read Creating your own Suggestion Analyzer.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="select label"  name="okp:consulting.select" type="multiple"
            suggestion="com.openkm.form.suggestion.DocumentContentContainsSuggestion"
      <option label="one" value="001" />
      <option label="two" value="002" />
      <option label="three" value="003" />
     </select>
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Separator field

Draw a separator between fields.

Attribute	*Description*	*Required*
label	*The text shown as a label in the user interface.*	*true*
name	*Unique field identifier* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "****okp:****".* *It's a good practice to use a metadata group name as basis for metadata field name. For example use "****okg:consulting****" has been used for field name "****okp:consulting****.separator", changin "****okg****" to "****okp****". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
readonly	*The set field will not be able to be modified in the user interface. By default the value is "* ***false****".*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Input label" name="okp:consulting.input1" />
    <separator label="Separator label" name="okp:consulting.separator" />
    <input label="Input label" name="okp:consulting.input2" />
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Suggestbox field

This is an implementation of the typical AJAX suggest box component:

Attribute	*Description*	*Required*
label	*The text shown as a label in the user interface.*	*true*
name	*Unique field identifier.* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use a metadata group name as basis for metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.suggestbox", changin "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
description	*General description.*	*false*
filterQuery	*A SQL query to filter the data. Use {0} to place the user input.*	*
valueQuery	*A SQL query to obtain the value of a identifier. Use {0} to place the identifier.*	*
className	*A class used to get the selected options.*	*
filterMinLen	*The minimum chars in input to begin to filter results.*	*true*
dialogTitle	*The title of the HTML dialog.*	*true*
readonly	*The set field will not be able to be modified in the user interface. By default, the value is*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	*"**false**".*	
dbColumnSize	*Set the size of the database field where the value is stored.* *By default, fields are stored in a database column with 128 characters size. Because your data length might be bigger than 128 characters the administrator should update the default column size.* *Take a look at Metadata XML definition documentation section for a deep explanation about metadata storage in OpenKM database,*	

Child elements:

- *Validator. For more information see Metadata Validator element.*

** There're three options to get suggestbox values (**use one of them is mandatory**):*

- *filterQuery + valueQuery*
- *className*

No other combination of them is allowed.

Based on filterQuery and valueQuery

*Now **filterQuery** and **valueQuery** could be a regular SQL query.*

You should only use {0} as the parameter entered by the user to filter the values from the table.

Create COUNTRY table:

```
create table COUNTRY (
  CT_ID varchar(2),
  CT_NAME varchar(32),
  primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Metadata definition:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
  <suggestbox label="country" name="okp:consulting.suggestbox"
     width="200px" dialogTitle="Choose country" filterMinLen="3"
    filterQuery="select ct_id, ct_name from country where ct_name like '%{0}%' order
    valueQuery="select ct_id, ct_name from country where ct_id='{0}'" />
  </property-group>
</property-groups>
```

Based on className

To help to get a list from external applications or other purposes, your own class can be built to retrieve the option values from those systems.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
  <suggestbox label="country" name="okp:consulting.suggestbox"
     width="200px" dialogTitle="Choose country" filterMinLen="3"
    className="com.openkm.plugin.form.values.SuggestBoxUserList" />
  </property-group>
</property-groups>
```

For more information read Creating your own Suggestbox plugin

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Text field

Prints a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique field identifier* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "****okp:****".* *It's a good practice to use metadata group name as basis for metadata field name. For example use "****okg:consulting****" has been used for field name "****okp:consulting****.text", changin "****okg****" to "****okp****". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
readonly	*The set field will not be able to be modified in the user interface. By default the value is "* ***false****".*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <text label="text label" name="okp:consulting.text" width="200px" />
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata TextArea field

Prints a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as a label in theuser interface.*	*true*
name	*Unique field identifier* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "****okp:****".* *It's a good practice to use metadata group name as basis for metadata field name. For example use "****okg:consulting****" has been used for field name "****okp:consulting****.textarea", changin "****okg****" to "****okp****". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
value	*Default value for the component.*	*false*
placeholder	*The placeholder value.*	*false*
description	*General description.*	*false*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
readonly	*The set field will not be able to be modified in the user interface. By default the value is "****false****".*	*false*
dbColumnSize	*Set the size of the database field where the value is stored.* *By default, fields are stored in a database column with 128 characters size. Because your data length might be bigger than 128 characters the administrator should update the default column size.*	

Take a look at Metadata XML definition documentation section for a deep explanation about metadata storage in OpenKM database,

Child elements:

- *Validator. For more information see Metadata Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <textarea label="textarea label" name="okp:consulting.textarea" width="200px" />
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Validator element

The goal of the Validatior element is to ensure that the user provides necessary and properly formatted information needed to successfully complete an operation.

The validator can only be used in combination with other fields:

- *Metadata Checkbox field*
- *Metadata Input field*
- *Metadata Select field*
- *Metadata Suggestbox field*
- *Metadata TextArea field*

Available validators:

name	*Description*
req	*Required value.*
alpha	*Alphanumeric value.*
dec	*Decimal value.*
num	*Numeric value.*
email	*E-mail value.*
url	*URL value.*
maxlen	*Set the maximum length allowed.*
minlen	*Set the minimum length required.*
lt	*Less than.*
gt	*Greater than.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

min	*Minimum value.*
max	*Maximum value.*
regexp	*Regular expression.*
plugin	*Creates your own form validator.* *More information at Creating your own Form Validator plugin.*

Parent:

- *Metadata Checkbox field*
- *Metadata Input field*
- *Metadata Select field*
- *Metadata Suggestbox field*
- *Metadata TextArea field*

Basic example

- *Value required "**req**".*
- *Value must be decimal "**dec**".*
- *The maximum lenght of the decimal value is 6 digits "**maxlen**".*

```
<input label="Decimal value" name="okp:consulting.decimal">
    <validator type="dec" />
    <validator type="req" />
    <validator type="maxlen" parameter="6" />
</input>
```

Regular expression

If you want a validation which is not included in these prefefined, you can use a regular expression. For example, imagine you want to restrict an input to lowecase ASCII values:

```
<input label="Lowercase value" name="regExp">
    <validator type="regexp" parameter="[a-z]+" />
</input>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Option element

The option element is only used in combination with Metadata Select field. The option value represents a list of distinct values to select.

Attribute	*Description*	*Required*
label	*The text shown as label in the user interface.*	*true*
value	*Unique value identifier.* *Two options can't have the same value. **Value must be unique**.* *It's recommended to only use characters which matches this regular expression: [a-z_] (a to z in lowercase and underline, no spaces)*	*true*
parentValue	*When present, it specifies the option parent.* *The parent must be present in other select.*	*false*
selected	*When present, it specifies that an option should be pre-selected when the form is redered.*	*false*
cssClass	*CSS class.* *More information about creating your own styles at CSS.*	*false*

Parent:

- *Select. For more information, Please see: Metadata Select field.*

Basic select with selected option

- *attribute selected=true, when present, it specifies that an option should be pre-selected when the page loads.*

```
<?xml version="1.0" encoding="UTF-8"?>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="select label"  name="okp:consulting.select" type="simple">
      <option label="one" value="001" />
      <option label="two" value="002" selected="true"/>
      <option label="three" value="003" />
     </select>
  </property-group>
</property-groups>
```

Hierarchical relationship

- *The field option value of "**okp:consulting.children**" have a parent relation with "**okp:consulting.parent**". It's defined by **parentElement="okp:consulting.parent"**.*
- *The option value "**benefits**" of the field "**okp:consulting.children**" has a parent value "**directives**" from field "**okp:consulting.parent**".*

Tree relationships:

- *directives*
  - *benefits*
  - *elegibility*
  - *services*
- *handbook*
  - *basic*
  - *normal*
  - *advanced*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <select label="Provincia" name="okp:consulting.parent" type="simple">
      <option value="directives" label="Directives" />
      <option value="handbook" label="Handbook Paragraphs" />
    </select>
    <select label="Provincia" name="okp:consulting.children" type="simple" parentElem
      <option value="benefits" label="Benefits" parentValue="directives" />
      <option value="elegibility" label="Eligibility" parentValue="directives" />
      <option value="services" label="Services" parentValue="directives" />
      <option value="basic" label="Basic" parentValue="handbook" />
      <option value="normal" label="Advanced" parentValue="handbook" />
      <option value="advanced" label="Normal" parentValue="handbook" />
    </select>
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata Input field

In most cases this field is used to enter free form text, althought it can be used to save http links, date, data or refer to application folder (as internal link to this folder).

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique field identifier.* *Two metadata can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use metadata group name as basis for metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.input", changin "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
type	*Type value. Allowed types:* • *text.* • *date.* • *link.* • *folder.*	*true*
value	*Specifies a default value for this component.* *There's a special case for date types. Data is stored in format "**yyyyMMddHHmmss**", be aware of this format when using OpenKM API.*	*false*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
placeholder	*The placeholder value.*	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

description	*General description.*	*false*
readonly	*The set field will not be able to be modified in the user interface. By default the value is "**false**".*	*false*
timeFormat	*When input type is date we can specify if want to be able to see and modify hours and minutes. Allowed values:* • *hm* • *h* • *none*	*false*
dbColumnSize	*Set the size of the database field where the value is stored.* *By default, fields are stored in a database column with 128 characters size. Because your data length might be bigger than 128 characters the administrator should update the default column size.* *Take a look at Metadata XML definition documentation section for a deep explanation about metadata storage in OpenKM database,*	

Child elements:

- *Validator. For more information please see Metadata Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Input label" name="okp:consulting.input" />
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata VerticalPanel field

Vertical layout that allow you to sort components inside.

Attribute	*Description*	*Required*
name	*Unique field identifier* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use a metadata group name as basis for metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.separator", changin "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <vpanel name="okp:consulting.vertical1">
       <input label="Input label" name="okp:consulting.input1" />
    </vpanel>
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata HorizontalPanel field

Horizontal layout that allow you to sort components inside.

Attribute	*Description*	*Required*
name	*Unique field identifier* *Two metadatas can't have the same name.* ***Name must be unique****.* *The name must start with "**okp:**".* *It's a good practice to use a metadata group name as basis for metadata field name. For example use "**okg:consulting**" has been used for field name "**okp:consulting**.separator", changin "**okg**" to "**okp**". That helps to find out the metada group based on metadata field name.* *Please, use only letters, numbers and underscore: "0-9a-zA-Z_".*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <hpanel name="okp:consulting.horizontal1">
       <input label="Input label" name="okp:consulting.input1" />       <input label=
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Manage users and roles

In the Manage users and roles view you can do:

- *Manage users*.
- *Manage roles*.
- *Manage user mail accounts*.
- *Logged users*
- *Message queue*

By default the application manages users and roles in its own database.

It is also possible to connect the application to an external directory (LDAP directory for instance).

Also the login module can be extended and create your own login adapter. More information about it Creating your own login module.

When the application is connected to LDAP the manage users and roles view is automatically set to readonly mode.

In this scenario the LDAP takes full control of users and roles.

The restricted actions are:

- *Create, delete or modify a user.*
- *Create or delete a role.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Manage users

To manage users *on the top right click on* ***User list button***.

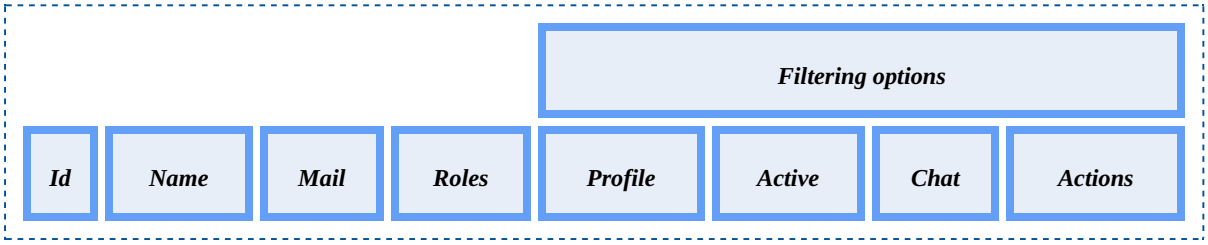
By default the application manages users and roles in its own database.

It is also possible to connect the application to an external directory (LDAP directory for instance).

The login module can be extended and create your own login adapter. More information about it *Creating your own login module*.

When the application is connected to LDAP the manage users view is automatically set to readonly mode. Almost all actions are restricted.

Users are shown in a table structure.



User parameters:

Field	*Description*	*Required*
Id	*The id the user will use to log in.* *Two users can't have the same id.*	*true*
Name	*User name.*	*true*
Mail	*User email address.*	*true*
Roles	*Roles - groups - from which the user is a member.* *User must be at least in 'ROLE_ADMIN' or 'ROLE_USER' role.*	*true*
Profile	*Each profile is a set of options available on the user interface.*	*true*

Active	*Sets user enabled or not.*	*true*
Chat	*Indicates if user is connected to the chat.* *Althought a user can have the application opened on several browser tabs. Only one of them will be logged on chat.*	*false*
Action	*Available actions:* • *Edits a user.* • *Deletes a user.* • *Sets the profile.* • *Configures mail accounts. More information here Manage user mail accounts.* • *Configures twitter account.* • *Shows the activity log of the user.* • *Delete 2FA user secret*	*
** Mail account and twitter account are optional.*		

Create a user

- *On bottom right, click on the* ***New user icon***.
- ***Fill all the fields***. *Remember all users must be at least in 'ROLE_ADMIN' or 'ROLE_USER' roles.*
- *Click on the* ***Create button***.

Edit a user

- *Click on the* ***Edit user icon***.
- ***Modify the fields***.
- *Click on the* ***Edit button***.

Delete a user

- *Click on the* ***Delete user icon***.
- *Click on the* ***Delete button***.

Change the user profile

- *Click on the* ***User config icon***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Choose the profile.*
- *Click on the **Edit button**.*

By default the profile named "default" is automatically assigned to any new user.

Export the users

- *On the top right click on the* ***export users**.*
- *The application will download a .csv file.*

Enable twitter account

- *Click on the* ***Configure twitter account**.*
- *Set **twitter user**.*
- ***Enable active** checkbox.*
- *Click on the **Create button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Manage user mail accounts

In the user acounts view, are shown the mail user accounts. These mail accounts are used by application to periodically import mail to application repository.

***To manage user mail account**, **from Users view** click on* ***Email account icon**.*

Mail account hicheracy

- *Mail account*
  - *Filter*
    - *Rules*

- *The mail account sets the data connection to the mail server to retrieve mails.*
- *Filters are optional and sets the appliation location - folder - where to store mails.*
- *Rules are optional and sets a couple of rules that controls what are the valid mails to import.*

When a filter is not set, by default the application will store mail into /okm:mail/userid folder.

When rules are not set, by default any incoming mail is accepted.

Mail accounts are shown in a table structure.

Mail account parameters:

Field	*Description*	*Required*
Mail protocol	*The protocol used to connect to the mail server.* *Available options:* • *IMAP* • *IMAPS* • *POP3* • *POP3S*	*true*
Mail host	*The mail server host.*	*true*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Mail user	*The user name to access the mail account.*	*true*
Mail password	*The password to access the mail account.*	*true*
Mail folder	*The IMAP folder.* *Only applicable when mail protocol value is IMAP or IMAPS. In case of POP3 you can leave empty otherwise need to be set to INBOX.*	*false*
Mail mark seen	*Sets to mail server the seen status.* *Available options:* • *Keep (keep as is).* • *Yes (mark as seen)* • *No (mark as not seen).*	*true*
Mail mark deleted	*Set mail to be deleted.* *When checked the mail will be marked to be deleted from the mail server.*	*true*
Active	*Sets mail account enabled or not.*	*true*
Actions	*Available actions:* • *Edits a mail account.* • *Deletes a mail account.* • *Sets mail account configuration filters.* • *Browsers emails in the mail server.*	

Create a new mail account

- *At the top right click on the* ***Create new account icon***.
- ***Fill in all the fields***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on the **Create button**.*

> *It is a good practice before clicking on Create button, to **check the mail configuration**. There's a button named **Check** that will check the mail configuration. If all is right application will get **OK** status.*

Edit a mail account

- *Click on the* ***Edit mail account icon**.*
- ***Modify the fields**.*
- *Click on the **Edit button**.*

Delete a mail account

- *Click on the* ***Delete mail account icon**.*
- *Click on the **Delete button**.*

Browser emails in the mail server

- *Click on the* ***Server list icon**.*

> *In this view mail can be directly imported.*

Manage mail account filters

In the mail account configuration, rules can be set to automatically store mails on different application folders.

> *For example, filters can help to automatically catalog mails, based on a "subject" value or "from" value among others.*

Mail accounts filter are shown in a table structure.

Folder	***Grouping***	***Active***

Mail account filter parameters:

Field	*Description*	*Required*
Folder	*Sets the destination folder.*	*true.*
	*Sets automatic catalog by **year/month/day** enabled.*	

Grouping	*To prevent having too many mails in the same folder, we recommend enabling mail grouping.* *The folders year/month/years are automatically created by application.*	*false.*
Active	*Sets filter enabled or not.*	*true.*
Actions	*Available actions:* • *Edit a filter.* • *Delete a filter.* • *Show rules.*	

At least a filter sets a destination folder it also can have additional rules.

A rule can only be set into a filter.

Mail accounts rules are shown in a table structure.

Field	***Operation***	***Value***	***Active***

Mail account rule parameters:

Field	*Description*	*Required*
Field	*Sets the mail field that will be evaluated by the rule.* *Available options are:* • *From* • *To* • *Subject* • *Content*	*true.*
Operation	*Set the operation that will be done for the evalution process.* *Available options are:* • *Contains*	*true.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	• *Equal*	
Value	*Sets the value that will be used for evaluation process.*	*true.*
Active	*Sets rule enabled or not.*	*true.*

Create a mail account filter

- *Click on the* ***Show filters icon***.
- *On top right click on* ***New filter icon***.
- ***Fill in all the fields***.
- *Click on the* ***Create button***.

Edit a mail account filter

- *Click on the* ***Show filters icon***.
- *On top right click on* ***Edit filter icon***.
- ***Modify the fields fields***.
- *Click on the* ***Edit button***.

Delete a mail account filter

- *Click on the* ***Show filters icon***.
- *On top right click on* ***Delete filter icon***.
- *Click on the* ***Delete button***.

Create a mail account rule

- *Click on the* ***Show filters icon***.
- *Click on the* ***Show rules icon***.
- *On top right click on* ***New rule icon***.
- ***Fill in the fields.***
- *Click on the* ***Create button***.

Edit a mail account rule

- *Click on the* ***Show filters icon***.
- *Click on the* ***Show rules icon***.
- *On top right click on* ***Edit rule icon***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Modify the fields.***
- *Click on the* ***Edit button***.

Delete a mail account rule

- *Click on the* ***Show filters icon***.
- *Click on the* ***Show rules icon***.
- *On top right click on* ***Delete rule icon***.
- *Click on the* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Manage roles

To manage roles *at the top right click on* ***Role list button***.

By default the application manages users and roles in its own database.

It is also possible to connect the application to an external directory (LDAP directory for instance).

Also the login module can be extended and create your own login adapter. More information about it *Creating your own login module*.

When the application is connected to LDAP the manage roles view is automatically set to readonly mode. Almost all actions are disabled.

Roles are shown in a table structure.

Id	***Active***	***Actions***

Role parameters:

Field	*Description*	*Required*
Id	*The id of the role.* *Two roles can't have the same id.*	*true*
Active	*Sets if role is enabled or not.*	*true*
Action	*Available actions:* • *Edits a role.* • *Deletes a role.*	

Create a role

- *At the bottom right, click on the* ***New role icon***.
- ***Fill all the fields***.
- *Click on the* ***Create button***.

Edit a role

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on the* ***Edit role icon***.
- ***Modify the fields***.
- *Click on the* ***Edit button***.

ROLE id can't be modified to prevent repository security inconsistences. This operation can be done only at database level.

Delete a role

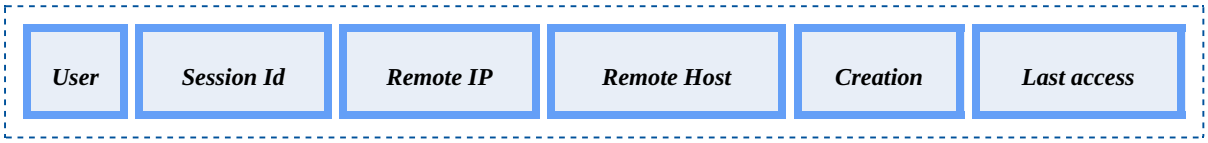
- *Click on the* ***Delete role icon***.
- *Click on the* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Logged users

To manage logged users *on top right click on* ***Logged users list button***.

Logged users are shown in a table structure :



Real time user activity log

- *Click on on* ***Activity log icon***.
- *Will be shown as a table with all actions performed by the user today.*
- *Click on the* ***Filter button*** *to refresh values.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Message queue

Message queue helps administrators to send messages across the user interface.

Helps administrators to send a message to all logged users to indicate the system will be halt and automatically logout the users 5 minutes after they received the message.

The application sends messages to a user when a background process has been completed.

The application sends warnings to a user about an action that is not allowed.

Among others messages.

Queue of messages shown in a table structure:

Message parameters:

Field	*Description*	*Required*
Date	*Date of the message.* *Automatically set by the application.*	*true*
Action	*Available values:* • *Logout (force users who receives it to logout)*	*false*
Message	*The text of the message.*	*true*
To All	*When present, the message is sent to all the users.*	*
To Users	*When present, indicate a set of users to sent the message.*	*
To Roles	*When present, indicate a set of users which are members of some of these roles.*	*

** At least one of the options.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Send a message

- *Click on the* ***New message icon***.
- ***Fill the fields***.
- *Click on the* ***Create button***.

Edit a message

- *Click on the* ***Edit message icon***.
- ***Modify the fields***.
- *Click on the* ***Edit button***.

Delete a message

- *Click on the* ***Delete message icon***.
- *Click on the* ***Delete button***.

Delete all messages queue

- *On top right click on* ***Clear button***.

This operation can not be reverted. All the message queue will be deleted.

Refresh message queue

- *On the top right click on* ***Refresh button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profiles

Profiles are a set of user interface configuration parameters. They help the administrators to set the available interface features for each user.

For example:

- *Restricts the user interface hidding search and dashboard tabs and almost toolbar buttons.*
- *Restricts the user interface showing only search view.*

The application and specially the user interface comes by default with a several features and extensions.

Not all users need access to everything, specially at the begining when a lot of features can cause confusion to end user.

Our suggestion is going from less to more features as a natural process of the application knowledge.

Profiles are shown in a table structure.

Name	Active	Actions

Profiles are divided in five tabs:

- *Profile General tab*

Create a new profile

- *Click on the* ***New profile icon***.
- ***Fill in the fields***.
- *Click on the* ***Create button***.

Edit a profile

- *Click on the* ***Edit profile icon***.
- ***Modify the fields***.
- *Click on the* ***Edit button***.

Delete a profile

- *Click on the* ***Delete profile icon***.
- *Click on the* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Clone a profile

- *Click on the* ***Clone profile icon***.
- ***Modify the fields***.
- *Click on the* ***Edit button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profile General tab

General section

Field	Description
Name	*The name of the profile.* *Two profiles cannot share the same name.*
Active	*Sets profile enabled or not.*

Misc section

Field	Description
Quota limit (MB)	*Sets the total maximum size of documents that a user can upload.* *Value 0 indicate unlimited.* *Helps the administrator to take control of the repository growth, and detect user's abuse.*
Advanced filters	*Enables search filtering on users and roles popup view.* *Helps to find roles or user in a large list.*
Web skin	*Sets the default skin (css styles applied).* *Info message*
Print preview	*Enables or disables printing funtion.*
Keywords enabled	*Enables or disables keywords shortcuts.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Upload notify users	*Shows or hides user notification on upload popup.*
Notify external users	*Enables or disables send notifications to external users.* *By default the application is restricted to mail user's addresses.* *For example, sending documents to an external mail address.*
Browser plugin preview	*When enabled, preview is based on browser pdf render capability.*
Browser HTML5 preview	*Enables or disables preview on HTML5 format.*
Increase version	*When enabled, users can choose between document upload as minor version or major version.*
Add documents title	*When enabled, documents have an extra field named title.* *Is shown a title on :* • *A column in a file explorer view.* • *Search view* • *When uploading a new document.* • *On document properties tab.*
Show events notification	*When enabled the user shows in a popup messages sent by application.* *Common notifications:* • *New documents uploaded.* • *New task assigned.* • *New mails received.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Thumbnail visible	*When enabled a thumbnail is shown on the desktop in search view.* *This feature can decrease the application performance.* *For more information see Performance configuration parameters.*
Sent Mail Storage	*Set default mail storage location.* *Available options:* • *Mail folder ("**/okm:root/${user}/sent**")* • *Current folder (user selected frontend UI folder).*
Quick search visible	*Show or hide top right quick search in UI.*
Recursive security change	*Enable or disable by default the checkbox for applying security changes recursively.*
Copy related information	*Enable or disable copy attributes while creating documents from templates.*
Extensions	*Sets the enabled extensions for the profile.*
Reports	*Sets the enabled reports for the profile.*
Workflows	*Sets the available workflows for the profile.*
Property Groups	*Sets the available property groups for the profile.*

Chat section

Field	*Description*
Enabled	*Enables or disables chat.*

Auto login	*When enabled the user is automatically logged in chat as part of login process.*

Wizard section

The Wizard only takes effect when uploading a new document, not when updating and existing.

In case an automation definition is present, these parameters has no effect.

The automation feature covers all these cases and more.

Field	*Description*
Metadata	*Sets a metadata that will be required to be filled as part of uploading a new document.*
Workflow	*Sets a workflows that will be executed as part of uploading a new document.* *Some workflows require initial fields filled at the begining, but not all.*
Keywords	*When enabled the application will ask user to add keywords as part of uploading a new document.* *User can skip this step without adding any keyword.*
Categories	*When enabled the application will ask user to add categories as part of uploading a new document.* *User can skip this step without adding any category.*
OCR Data Capture	*When enabled the application will try to capture data with OCR feature as part of uploading a new document.* *Application execute OCR data capture only from images and pdf MIME types.*
Cancel wizard	*When enabled, a button that allows the user to cancel operation is shown in Wizard popup .*

Pagination section

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Pagination control

Field	*Description*
Enabled	*Enables or disable pagination on file explorer view.*
Page list	*Sets the distinct values in what will be shown in number of elements per page.* *For example the value "10;25;50;100" set that maximum number of element per page can be 10, 25, 50 or 100.* *On the user view, the user can choose one of these options.* *When pagination is enabled at least one value must be set* *To separate the distinct values must be used the caracter ";"*
Type filter	*When enabled a filter by type is shown, this helps the user to show only documents, folders, records or mails at the same time. **|
Misc filter	*Advanced filter helps on filtering by some fields:* • *name* • *title* • *range of sizes* • *range of dates* • *author* • *version* *When extra metadata columns are show in file explorer, they are also available here.*
Show folders by default	*By default shows or hides folders. **|
Show documents by default	*By default shows or hides document. **|

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Show mails by default	*By default shows or hides mails. **
Show records by default	*By default shows or hides records. **

** When type filter is enabled the options show folders by default etc... they must be active, otherwise they are not taken in consideration and all types are shown.*

Section General

Shortcuts are keywords combination for faster way of accomplishing a task.

Field	*Description*
Rename	*Executes rename action on the selected node.* `F2`
Copy	*Executes copy action on the selected node.* `CTRL+SHIFT+C`
Cut	*Executes cut action on the selected node.* `CTRL+SHIFT+X`
Paste	*Executes paste action on the selected node.* `CTRL+SHIFT+V`
Download	*Executes download action on the selected node.* `CTRL+D`
Testing	*Shows a popup to do some user interface test.* `CTRL+G`
Log	*Shows user interface log.*

	`CTRL+Z`
Create Folder	*Executes create a folder action on the selected node.* `CTRL+N`
Refresh	*Executes refresh action.* `F5`
Add Document	*Executes add document action.* `INSERT`
Bookmarks	*Executes a bookmark action on the selected node.* `ALT+B`
Help	*Shows a help popup.* `F1`
Logout	*Executes the application logout.* `CTRL+ALT+Q`
Delete	*Executes a delete action on the selected node.* `ALT+DELETE`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profile Components tab

Stacks section

Field	*Description*
Taxonomy	*Shows or hides taxonomy in the stack panel.*
Categories	*Shows or hides categories in the stack panel.*
Thesaurus	*Shows or hides the thesarus in the stack panel.*
Templates	*Shows or hidesthe templates in the stack panel.*
Personal	*Shows or hides personal in the stack panel.*
Mail	*Shows or hides mail in the stack panel.*
Metadata	*Shows or hides metadata in the stack panel.*
Trash	*Shows or hides trash bin in the stack panel.*

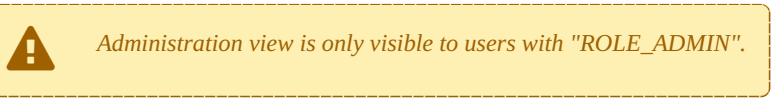
Dashboard section

Field	*Description*
User	*Shows or hides the user panel on dashboard view.*
Mail	*Shows or hides the new mail panel in the dashboard view.*
News	*Shows or hides the news panel in the dashboard view.*
General	*Shows or hides the general panel in the dashboard view.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow	*Shows or hides the workflow panel in the dashboard view.*
Keywords	*Shows or hides the tag cloud panel in the dashboard view.*
Mail Account	*Shows or hides the user mail account panel in the dashboard view.*
Task Manager	*Shows or hides the task manager panel in the dashboard view.*
Calendar	*Shows or hides the calendar panel in the dashboard view.*

Tab section

Field	*Description*
Default tab	*Sets the default tab visible when user login the application.* *Available options:* • *Desktop* • *Search* • *Dashboard* • *Administration* *Must set an enabled view.*
Desktop	*Shows or hides the desktop view.*
Search	*Shows or hides the search view.*
Dashboard	*Shows or hides the dashboard view.*
Administration	*Shows or hides the administration view.*  *Administration view is only visible to users with "ROLE_ADMIN".*

Tab document section

The tab document is visible when a document is selected in the desktop view.

Field	Description
Properties	Shows or hides the document properties tab.
Security	Shows or hides the security tab.
Notes	Shows or hides the notes tab.
Versions	Shows or hides the versions tab.
Version download	Enables or disables the download document version.
Preview	Shows or hides the preview tab.
Metadata	Shows or hides the metadata tabs.
Relation	Shows or hides the relations tab.

Tab folder section

The tab folder is visible when a folder is selected in desktop view.

Field	Description
Properties	Shows or hides the folder properties tab.
Security	Shows or hides the folder tab.
Notes	Shows or hides the notes tab.
Preview	Shows or hides the preview tab.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata	*Shows or hides the metadata tabs.*
Relation	*Shows or hides the relations tab.*

Tab mail section

Tab mail is visible when a mail is selected in desktop view.

Field	*Description*
Properties	*Shows or hides the mail properties tab.*
Security	*Shows or hides the mail tab.*
Notes	*Shows or hides the notes tab.*
Preview	*Shows or hides the preview tab.*
Metadata	*Shows or hides the metadata tabs.*
Relation	*Shows or hides the relations tab.*

Tab record section

The tab record is visible when a record is selected in desktop view.

Field	*Description*
Properties	*Sets the default search tab visible when a user login the application..*
Security	*Shows or hides the record tab.*
Notes	*Shows or hides the notes tab.*
Preview	*Shows or hides the preview tab.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata	*Shows or hides themetadata tabs.*
Relation	*Shows or hides the relations tab.*

Tab search section

Field	*Description*
Default search	*Sets the default tab visible in search view.* *Available options:* • *Simple* • *Advanced*
Simple	*Shows or hides the simple search view.*
Advanced	*Shows or hides the advanced search view.* *Advanced search view is composed of:* • *Tab basic* • *Tab advanced* • *Tab metadata*
Tab basic	*Shows or hides the basic tab in advanced search view.*
Tab advanced	*Shows or hides the advanced tabs in the advanced search view.*
Tab metadata	*Shows or hides the metadata tab in the advanced search view.*
Export CSV	*Shows or hides the export to csv feature in advanced search view.*
Export ZIP	*Enabled or disables the export to zip feature in the advanced search view.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Tab basic section

The tab basic contains the most common fields for searching.

Field	Description
Context	*Shows or hides the context field.*
Content	*Shows or hides the content field.*
Name	*Shows or hides the name field.*
Title	*Shows or hides the title field.*
Keywords	*Shows or hides the keywords field.*
Language	*Shows or hides the language list.*
User	*Shows or hides the user list.*
Date	*Shows or hides the date range fields.*

Tab advanced section

Tab advanced contains the advanced fields for doing searching.

Field	Description
Folder	*Shows or hides the folder selector.*
Category	*Shows or hides the category selector.*
Type	*Shows or hides the node type.* *In search view four types are shown:* • *Document*

	• *Folder* • *Mail* • *Record*
Document type	*Shows or hides the document type list.* *Contains a list of MIME type.* *To set the element to be shown in the list see MIME types*
Notes	*Shows or hides the the notes field.*
Mail fields	*Shows or hides the mail fields.* *Available mail fields:* • *Email from.* • *Email to.* • *Email subject.*

Profile Menu tab

These menu options have preference among toolbar options or contextual menu options.

Hierarchy:

- *Menu*
  - *Toolbar*

Preference hierarchy. Upper hierarchy takes control over the lower. If the upper option is disabled then it will be also disabled at the lower one.

For example, when download menu option is disabled then it will not be shown at the toolbar althought there will be enabled.

Menu section

These are the main menu options.

Field	*Description*
File	*Shows or hides the file menu.*
Edit	*Shows or hides the edit menu.*
Tools	*Shows or hides the tools menu.*
Bookmarks	*Shows or hides the bookmark menu.*
Templates	*Shows or hides the templates menu.*
Help	*Shows or hides the help menu.*

Menu bookmark section

Field	*Description*
Manage bookmarks	*Shows or hides manage the bookmarks menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Add bookmark	*Show or hide add the bookmark menu option.*
Set home	*Shows or hides the set home menu option.*
Go home	*Shows or hides the go to home menu option.* *On loading, the application always try to open the default home.* *The default home is /okm:root that can be changed by end user.*

Menu tool section

Field	*Description*
Languages	*Shows or hides the change language menu option.*
Skin	*Shows or hides the change skin menu option.*
Debug	*Shows or hides the debug menu option.* *When enabled is shown, it helps the administrator to find out possible UI bugs.*
Administration	*Shows or hides the administration menu option.* *Open administration is on a new window.*
Preferences	*Shows or hides the user preferences menu option.*
OCR Capture data	*Shows or hides the ocr data capture menu option.*
OMR	*Shows or hides the user omr - Object Mark Recognition - menu option.*
	Shows or hides the convert menu option.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Convert	*This option is used to convert documents to different formats.* *For example .doc to PDF.*
Web page import	*Shows or hides the web page menu option.*

Menu help section

Field	*Description*
Documentation	*Shows or hides the go to documentation website menu option.*
Bug tracking	*Shows or hides the go to bug tracking website menu option.*
Support	*Shows or hides the support menu option.*
Forum	*Shows or hides the go to public forum website menu option.*
Changelog	*Shows or hides the go to changelog menu option.*
Web site	*Shows or hides the go to openkm website menu option.*
About	*Shows or hides the application information menu option.*

Menu file section

Field	*Description*
Create folder	*Shows or hides the create folder menu option.*
Find folders	*Shows or hides the find folders menu option.*
Find documents	*Shows or hides the find documents menu option.*
Find mails	*Shows or hides the find mails menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Find records	*Shows or hides the find records menu option.*
Similar documents	*Shows or hides the find similar documents menu option.*
Go folder	*Shows or hides the go to folder menu option.* *This option for example is shown in the contextual menu at the search results.*
Download	*Shows or hides the download menu option.*
Download PDF	*Shows or hides the download as pdf menu option.*
Add document	*Shows or hides the add document menu option.*
Add record	*Shows or hides the add record menu option.*
Purge	*Shows or hides the purge menu option.*
Purge trash	*Shows or hides the purge all user trash menu option.*
Restore	*Shows or hides the restore from trash menu option.*
Start workflow	*Shows or hides the start workflow menu option.*
Refresh	*Shows or hides the refresh user interface menu option.*
Scanner	*Shows or hides the scanner tool menu option.*
Uploader	*Shows or hides the bulk upload menu option.*
Export	*Shows or hides the export to ZIP menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create from template	*Shows or hides the create from template menu option.*
Send document link	*Shows or hides the send document link menu option.*
Send document attachment	*Shows or hides the send document attachment menu option.*
Generate download link	*Shows or hides the generate download link menu option.*
Forward e-mail	*Shows or hides the forward email menu option.*
Write e-mail	*Shows or hides the write email menu option.*

Menu edit section

Field	*Description*
Lock	*Shows or hides the lock menu option.*
Unlock	*Shows or hides the unlock menu option.*
Update	*Shows or hides the update a document menu option.*
Edit	*Shows or hides the edit a document menu option.*
Cancel Edit	*Shows or hides the cancel edit a document menu option.*
Delete	*Shows or hides the delete menu option.*
Rename	*Shows or hides the rename menu option.*
Copy	*Shows or hides the copy menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Move	*Shows or hides the move menu option.*
Add subscription	*Shows or hides the add subscription menu option.*
Remove subscription	*Shows or hides the remvoe subscription menu option.*
Add metadata group	*Shows or hides the add metadata group menu option.*
Update metadata group	*Shows or hides the update metadata group menu option.*
Remove metadata group	*Shows or hides the remove metadata group menu option.*
Add note	*Shows or hides the add note menu option.*
Remove note	*Shows or hides the remove note menu option.*
Add category	*Shows or hides the add category menu option.*
Remove category	*Shows or hides the remove category menu option.*
Add keyword	*Shows or hides the add keyword menu option.*
Remove keyword	*Shows or hides the remove keyword menu option.*
Merge PDF	*Shows or hides the merge pdf documents menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profile Notification tab

The Application sends notifications based on events ocurred on user's subscriptions. These notifications can be sent as an immediate notification mail or all the messages grouped in a single mail delivered once a day.

Grouped notifications are sent by a crontab task name "Send Grouped Notifications".

Notifications are shown on a table structure:

Immediate	***Grouped***

*By default **these two groups are empty and no notifications will be sent**. If you want to work with notification you have to choose which events you want to be subscribed to.*

Events

Common

Notification	*Description*
COMMON	*Common event.*
COPY_ATTRIBUTES	*Copy attributes*

Document

Notification	*Description*
CANCEL_DOCUMENT_CHECKOUT	*Cancel the edition of a document.*
CHECKIN_DOCUMENT	*Update a document.*
CHECKOUT_DOCUMENT	*Mark the document for edition.*
COPY_DOCUMENT	*Copy a document.*
CREATE_DOCUMENT	*Create a new document.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

DELETE_DOCUMENT	*Delete a document.*
GET_CHILDREN_DOCUMENTS	*Get documents.*
GET_DOCUMENT_CONTENT	*Get the content of a document.*
GET_DOCUMENT_CONTENT_BY_VERSION	*Get the content of a document by version.*
GET_DOCUMENT_PROPERTIES	*Get the document properties.*
GET_DOCUMENT_VERSION_HISTORY	*Get the document version history.*
LOCK_DOCUMENT	*Lock a document.*
FORCE_LOCK_DOCUMENT	*Force lock a document.* *Available from OpenKM version 7.1.18*
MOVE_DOCUMENT	*Move a document.*
PURGE_DOCUMENT	*The document is definitely removed from repository.*
RENAME_DOCUMENT	*Change the name of a document.*
SET_DOCUMENT_PROPERTIES	*Change some document properties.*
UNLOCK_DOCUMENT	*Unlock a document.*
FORCE_ UNLOCK_DOCUMENT	*Force unlock a document.*
SET_DOCUMENT_NODE_CLASS	*Set the document NodeClass*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

SET_TITLE	*Set the title*
SET_DESCRIPTION	*Set the description*

Folder

Notification	*Description*
COPY_FOLDER	*Copy a folder.*
CREATE_FOLDER	*Create a new folder.*
DELETE_FOLDER	*Delete a folder.*
GET_CHILDREN_FOLDERS	*Get folders.*
GET_FOLDER_CONTENT_INFO	*Get the content info of a folder.*
GET_FOLDER_PROPERTIES	*Get the folder properties.*
MOVE_FOLDER	*Move a folder.*
PURGE_FOLDER	*The folder is definitely removed from repository.*
RENAME_FOLDER	*Change the name of a folder.*
SET_FOLDER_STYLE	*Set the folder style.*
SET_DESCRIPTION	*Set the description*

Mail

Notification	*Description*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CREATE_MAIL	*Create a new mail.*
GET_MAIL_PROPERTIES	*Get the mail properties.*
DELETE_MAIL	*Delete a mail.*
PURGE_MAIL	*The mail is definitely removed from repository.*
RENAME_MAIL	*Changes the name of a mail.*
MOVE_MAIL	*Move a mail.*
COPY_MAIL	*Copy a mail.*
GET_CHILDREN_MAILS	*Get mails.*
SET_MAIL_NODE_CLASS	*Set the mail NodeClass*
SET_DESCRIPTION	*Set the description*
SET_TITLE	*Set the title*

Record

Notification	*Description*
CREATE_RECORD	*Create a new record.*
GET_RECORD_PROPERTIES	*Get the properties of a record.*
DELETE_RECORD	*Delete a record.*
FORCE_LOCK_RECORD	*Force lock a record.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	Available from OpenKM version 7.1.18
FORCE_UNLOCK_RECORD	*Force unlock a record.* *Available from OpenKM version 7.1.18*
LOCK_RECORD	*Lock a record.*
PURGE_RECORD	*The record is definitely removed from the repository.*
RENAME_RECORD	*Changes the name of a record.*
UNLOCK_RECORD	*Unlock a record.*
MOVE_RECORD	*Move a record.*
COPY_RECORD	*Copy a record.*
GET_CHILDREN_RECORDS	*Get records.*
SET_RECORD_TITLE	*Set the record title.*
SET_RECORD_NODE_CLASS	*Set the record NodeClass*
SET_DESCRIPTION	*Set the description*
SET_TITLE	*Set the title*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profile Other tab

Toolbar section

Set the toolbar buttons that are shown.

> *To enable the toolbar option it must also be enabled in the menu tab, otherwise it will not be visible in the toolbar.*

Field	Description
Create folder	Shows or hides the create folder menu option.
Find folders	Shows or hides the find folders menu option.
Find documents	Shows or hides the find documents menu option.
Find mails	Shows or hides the find mails menu option.
Find records	Shows or hides the find records menu option.
Similar documents	Shows or hides the find similar documents menu option.
Download	Shows or hides the download document menu option.
Download PDF	Shows or hides the download document as pdf menu option.
Print	Shows or hides the print document menu option.
Lock	Shows or hides the lock menu option.
Unlock	Shows or hides the unlock menu option.
Add document	Shows or hides the add document menu option.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Add record	*Shows or hides the add record menu option.*
Write mail	*Shows or hides the write mail menu option.*
Edit	*Shows or hides the edit a document menu option.*
Update	*Shows or hides the update a document menu option.*
Cancel Edit	*Shows or hides the cancel edit a document menu option.*
Delete	*Shows or hides the delete menu option.*
Add metadata group	*Shows or hides the add metadata group menu option.*
Remove metadata group	*Shows or hides the remove metadata group menu option.*
Start workflow	*Shows or hides the start workflow menu option.*
Add subscription	*Shows or hides the add subscription menu option.*
Remove subscription	*Shows or hides the remove subscription menu option.*
Refresh	*Shows or hides the refresh user interface menu option.*
Home	*Shows or hides the go to home menu option.*
Scanner	*Shows or hides the scanner tool menu option.*
Uploader	*Shows or hides the bulk upload menu option.*
Splitter resize	*Shows or hides the splitter resize menu option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

History	*Shows or hides the history menu option.*
OMR	*Shows or hides the OMR - Object Mark Recognition - menu option.*

File browser section

Sets the file browser columns that are shown.

Field	*Description*
Status	*Shows or hides the status column.*
Massive	*Shows or hides the checkbox column.*
Icon	*Shows or hides the icon column.*
Name	*Shows or hides the name column.*
Title	*Shows or hides the title column.*
Language	*Shows or hides the language column.*
Size	*Shows or hides the size column.*
Last modified	*Shows or hides the last mofied date column.*
Author	*Shows or hides the author column.*
Version	*Shows or hides the version column.*
Column 1 to Colum 10	*Shows or hides the column based on metadata value.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Relation types

The application can sets distinct relations between nodes (documents, folders, mails, records). In relation types are set the relation types definitions.

There're tree types or relations:

- *Bidirectional*
- *Parent-child*
- *Many-to-many*

Relation types are shown in a table structure.

Type	Title	Title parent to child	Title child to parent

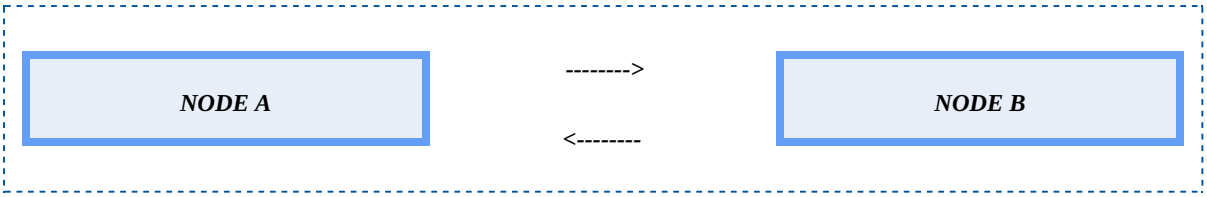
Types

Bidirectional

This releation is set always between two nodes.

A is linked to B like B is linked to A. The relation that links boths elements is the same.

An example of this relation type is "Language translation" relation. A is tranlations of B like B is translation of A.

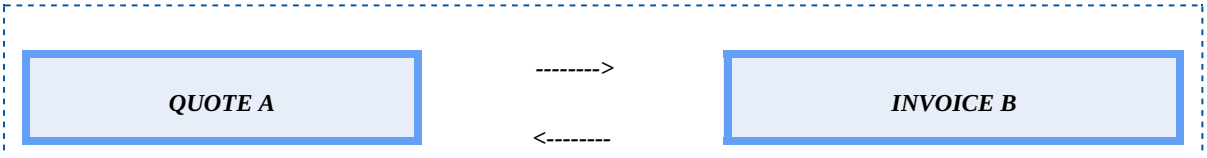


Parent - child

This relelation is always set between two nodes.

A is linked to B and B is linked to A, but the relation that links both elements is diferent.

An example of this relation type is "Invoice - Quote". B is an invoice of A. A is a quote of B.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Many-to-many

This relation is set between two or more nodes.

All nodes are linked to one another.

And example of this relation type is "Stapling". A,B,C, etc. are stapled.

Create a relation

- *On the top right click on* ***New relation type icon***.
- ***Choose one type*** *(bidirectional, parent-child or many-to_many).*
- ***Fill in the fields***.
- *Click on* ***Create button***.

On the bidirectional type is needed to fill the field "Title".

On the parent-child type are needed to fill the fields "Title parent to child" and "Title child to parent".

On many-to-many is needed to fill the field "Title".

Edit a relation

- *Click on* ***Edit a relation icon***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Delete a relation

Only relations that are not used can be deleted. If some nodes that are using that relation type exist, then the relation can not be deleted.

- *Click on* ***Delete a relation icon***.
- *Click on* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CSS

The application allowes to set your own CSS - Cascade Style Sheets - to be applied in user interface.

The CSS applied to user interface is set on Profile General tab section.

That helps the administrator to change the look and feel of the application.

For example, change the font size for cyrillic or logographic languages.

Css are shown in a table structure.

Name	Context	Active	Actions

Create a new CSS

Can not be two css with same name.

- *At the top right click on* ***New CSS icon***.
- ***Fill in the fields***.
  - *Name.*
  - *Context available are frontend or mobile.*
  - *Active.*
- *Click on* ***Create button***.

Edit a CSS

- *Click on* ***Edit CSS icon***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Delete a CSS

- *Click on* ***Delete CSS icon***.
- *Click on* ***Delete button***.

Clone a CSS

- *Click on* ***Clone CSS icon***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Modify the fields***.
- *Click on* ***Clone button***.

Download a CSS

- *Click on* ***Download CSS icon***.
- ***Elapsed some seconds, will be retrieved a CSS file***.

Samples

Large font sample

```
body, td, a, p, div, input, input.submit , textarea, select, button{
 font-family: Tahoma, Arial, Helvetica, sans-serif;
 font-size: 13px;
}

.gwt-MenuItem, .okm-TopMenuBar .gwt-MenuItem-selected, .okm-Input, .okm-TextArea, .ok
 font-family: Tahoma, Arial, Helvetica, sans-serif;
 font-size: 13px;
}

.okm-Popup .Caption, .okm-Popup-Error .Caption, .okm-Image-Popup .Caption {
 font-family: Tahoma, Arial, Helvetica, sans-serif;
 font-size: 15px;
}
```

Medium font sample

```
body, td, a, p, div, input, input.submit , textarea, select, button{
 font-family: Arial, Helvetica, sans-serif;
 font-size: 12px;
}

.gwt-MenuItem, .okm-TopMenuBar .gwt-MenuItem-selected, .okm-Input, .okm-TextArea, .ok
 font-family: Arial, Helvetica, sans-serif;
 font-size: 12px;
}

.okm-Popup .Caption, .okm-Popup-Error .Caption, .okm-Image-Popup .Caption {
 font-family: Arial, Helvetica, sans-serif;
 font-size: 14px;
}
```

Default font sample

```
body, td, a, p, div, input, input.submit , textarea, select, button{
 font-family: Arial, Helvetica, sans-serif;
 font-size: 11px;
}

.gwt-MenuItem, .okm-TopMenuBar .gwt-MenuItem-selected, .okm-Input, .okm-TextArea, .ok
 font-family: Arial, Helvetica, sans-serif;
 font-size: 11px;
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
.okm-Popup .Caption, .okm-Popup-Error .Caption, .okm-Image-Popup .Caption {
 font-family: Arial, Helvetica, sans-serif;
 font-size: 13px;
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Reports

The application comes with Jasper reports engine. That helps administrator to create own reports.

The JasperReports Library is one of the world's most popular reporting engine. It is able to use data coming from any type of data source and produce pixel-perfect documents that can be viewed, printed or exported in a variety of document formats including HTML, PDF, Excel, OpenOffice and Word.

More information about how to build your own reports at Creating your own reports.

More information about jasper report engine version that comes into application at Application version compatibility table.

*Keep on mind that any font used in the reports should be installed in the OpenKM server. For example, if you are using Arial (sample provided reports use this font) you should ensure Arial font is installed. For example, if you are using Ubuntu the package to install is **ttf-mscorefonts-installer**.*

```
$ apt-get install ttf-mscorefonts-installer
```

*In RedHat or CentOS the package is **msttcore-fonts**. You need to restart the server after this package is installed.*

If not sure about which fonts are available, execute this scripting code from Administration:

```
import java.awt.*;

for (String font : GraphicsEnvironment.getLocalGraphicsEnvironment().getAvailableFont
   print("Font: " +  font);
}
```

Reports are shown in a table structure.

Name	File name	Active

Create a new report

- *On the top right click on* ***New report icon***.
- ***Fill in the fields***.
- ***Choose a report file*** *(*.rep or *.jrxml)*
- *Click on* ***Create button***.

Edit a report

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on* ***Edit a report icon***.
- ***Modify the fields***.
- ***Optionally choose a report file*** *(*.rep or *.jrxml)*
- *Click on* ***Edit button***.

Delete a report

- *Click on* ***Delete a report icon.***
- *Click on* ***Delete button***.

Show report report parameters

> *Optionally, reports can require parameters for execution.*
>
> *For example a range of dates. These parameters will be shown as a form in the user interface when report will be executed.*

- *Click on* ***Report params icon.***

Execute a report

- *Click on* ***Execute report icon.***
- ***Fill in the fields*** *(when report comes with parameters).*
- *Choose report output format
  - *PDF format.*
  - *RTF format.*
  - *CSV format.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Reports pack

Most of these reports are based on Activity log events audit trail.

For more information Activity log.

Template reports

Name	Use Activity log	Require parameters	Description
JavaReportPattern.jrxml	No.	No.	Template pattern that use Java to retrieve data.
SQLReportPattern.jrxml	No.	No.	Template pattern that use SQL to retrieve data.

Authentication

Name	Use Activity log	Require parameters	Description
UsersLogin.rep	Yes.	Yes.	Users login, filtered by date range.
UsersLogout.rep	Yes.	Yes.	Users logout, filtered by date range.
UsersSessionExpiration.rep	Yes.	Yes.	Users that session expired, filtered by date range.

Documents

Name	Use Activity log	Require parameters	Description
DocumentCanceledCheckout.rep	Yes.	Yes.	Lists documents that cancelled the edition, filtered by date range.
DocumentCheckin.rep	Yes.	Yes.	List documents updated, filtered by date range.
DocumentCheckout.rep	Yes.	Yes.	List documents on edition, filtered by date

			range.
DocumentCreate.rep	*Yes.*	*Yes.*	*List documents created, filtered by date range.*
DocumentDelete.rep	*Yes.*	*Yes.*	*List documents deleted, filtered by date range.*
DocumentGetChildren.rep	*Yes.*	*Yes.*	*Show when the users listed the documents into some folder or record. Results are filtered by date range.* *To list documents into a folder is called internally method named "getChildren".*
DocumentContent.rep	*Yes.*	*Yes.*	*Show when the users retrieved the content of a document. Results are filtered by date range.* *For example, when a user download or preview a document (among other actions) is retrieving the content of the document.*
DocumentContentByVersion.rep	*Yes.*	*Yes.*	*Show when the users retrieved a specific version of the content of a document. Results are filtered by date range.*
DocumentGetProperties.rep	*Yes.*	*Yes.*	*Show when user retrieved the properties of a document. Results are filtered by date range.* *For example, when document is selected, is called internally a method named "getProperties" to fill the contents of properties tab in user interface.*
DocumentGetVersionHistory.rep	*Yes.*	*Yes.*	*Shown when the users retrieved the list of all*

			versions of a document. Results are filtered by date range.
DocumentGetPropertyGroupProperties.rep	*Yes.*	*Yes.*	*List when has been retrieved the fields of a metadata group.* *For example, when a metadata group is shown in users interface, is called internally a method named "getPropertyGroupProperties" to fill the metadata group values.*
DocumentLock.rep	*Yes.*	*Yes.*	*List documents updated, filtered by date range.*
DocumentMove.rep	*Yes.*	*Yes.*	*List documents moved, filtered by date range.*
DocumentPurge.rep	*Yes.*	*Yes.*	*List documents purged, filtered by date range.* *When a document is purged it will definitively disappear from application repository.*
DocumentRenamed.rep	*Yes.*	*Yes.*	*List documents renamed, filtered by date range.*
DocumentUnlock.rep	*Yes.*	*Yes.*	*List documents unlocked, filtered by date range.*
DocumentsAccessibleByUserOrdered.jrxml	*No.*	*No.*	*Retrieve a list of available documents - security filtered - ordered by folder name.* *Optimized for csv format.* *Not recommended for large repositories, can take a lot of time.*
DuplicatedFiles.jrxml	*No.*	*No.*	*List of duplicated files into application*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

			repository, based on document checksum.
ListDocumentsWithChecksum.rep	*No.*	*Yes.*	*List of all documents into some folders and subfolders with checksum value.*

Folders

Name	*Use Activity log*	*Require parameters*	*Description*
FolderCopy.rep	*Yes.*	*Yes.*	*List folders copyed, filtered by date range.*
FolderCreate.rep	*Yes.*	*Yes.*	*List folders created, filtered by date range.*
FolderDelete.rep	*Yes.*	*Yes.*	*List folders deleted, filtered by date range.*
FolderGetChilden.rep	*Yes.*	*Yes.*	*Show when the users listed the folders into some folder or record. Results are filtered by date range.* *To list folders into a folder is called internally method named "getChildren".*
FolderGetContentInfo.rep	*Yes.*	*Yes.*	*Shown when the internal method named "getContentInfo" has been executed.* *To do some operations is needed to calculate folder deep, for example before starting folder ZIP download.* *There's an internal method named "getCotentInfo" that does it.*
FolderGetProperties.rep	*Yes.*	*Yes.*	*Show when user retrieved the properties of a folder. Results are filtered by date range.* *For example, when folder is selected, is called internally a method named "getProperties" to fill the contents of properties tab in user interface.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FolderMove.rep	*Yes.*	*Yes.*	*List folders moved, filtered by date range.*
FolderPurge.rep	*Yes.*	*Yes.*	*List folders purged, filtered by date range.* *When a folder is purged it will definitively disappear from application repository.*
FolderRename.rep	*Yes.*	*Yes.*	*List folders renamed, filtered by date range.*
FolderSecurity.rep	*No.*	*Yes.*	*List folders and subfolders showing security regarding roles and/or users.*
FolderGetSize.rep	*No.*	*Yes.*	*List folders and subfolders showing the size of each one.*

Mails

Name	*Use Activity log*	*Require parameters*	*Description*
MailCreate.rep	*Yes.*	*Yes.*	*List mails created, filtered by date range.*
MailGetProperties.rep	*Yes.*	*Yes.*	*Shown when a user retrieved the properties of a mail. Results are filtered by date range.* *For example, when mail is selected, is called internally a method named "getProperties" to fill the contents of properties tab in user interface.*
MailPurge.rep	*Yes.*	*Yes.*	*List mails purged, filtered by date range.* *When a mail is purged it will definitively disappear from application repository.*
MailRename.rep	*Yes.*	*Yes.*	*List mails renamed, filtered by date range.*

MailMove.rep	*Yes.*	*Yes.*	*List mails moved, filtered by date range.*
MailCopy.rep	*Yes.*	*Yes.*	*List mails copyed, filtered by date range.*
MailGetChilden.rep	*Yes.*	*Yes.*	*Show when the users listed the mails into some folder or record.* *Results are filtered by date range.* *To list mails into a folder is called internally method named "getChildren".*

Others

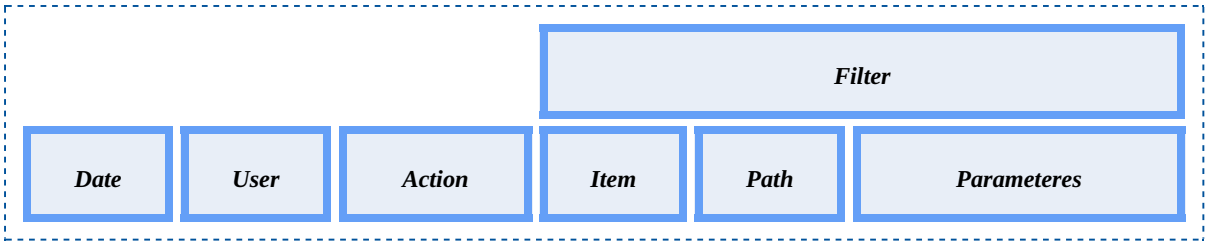
Name	*Use Activity log*	*Require parameters*	*Description*
TenantUserQuota.jrxml	*No.*	*No.*	*Document size (in MB) by user and tenant.*
WorkflowWorkload.jrxml	*No.*	*No.*	*Each workflow task with the time spent on it.*
RegisteredUsers.jrxml	*No.*	*No.*	*Registered users.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Activity log

This application performs a complete audit trail. Audit trail is mainly a datastore service. It defines a data record structure that will be used for storing audit information.

The activity log is shown in a table structure.



Filter options:

Field	Description	Required
From date	Start of date range.	true.
To date	End of date range.	true.
User	Specific user.	false.
Action	Any action.	false.

Auth actions

Field / Property	Description
LOGIN	User log in.
LOGOUT	User log out.
SESSION_EXPIRATION	User session expiration date
GRANT_USER	Adds user grant on a node.
REVOKE_USER	Removes user grant on a node.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

GRANT_ROLE	*Add role grant on a node.*
REVOKE_ROLE	*Removes user grant on a node.*
CHANGE_SECURITY	*Several security changes are applied in the same action.*

Document actions

Field / Property	*Description*
CANCEL_DOCUMENT_CHECKOUT	*Cancels document edition.*
CHECKIN_DOCUMENT	*Updates document.*
CHECKOUT_DOCUMENT	*Edits document.*
CREATE_DOCUMENT	*Creates a new document.*
DELETE_DOCUMENT	*Deletes a document.*
GET_CHILDREN_DOCUMENTS	*Retrieves a document list in a node.* *List of documents in a folder or record.*
GET_DOCUMENT_CONTENT	*Retrieves binary data.*
GET_DOCUMENT_CONTENT_BY_VERSION	*Retrieves binary data by document version.*
GET_DOCUMENT_PROPERTIES	*Retrieves all document properties.* *It does not include metadata. It has a direct relationship with the panel properties when a document is selected.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

GET_DOCUMENT_VERSION_HISTORY	*Retrieves a list of all versions of a document.*
LOCK_DOCUMENT	*Locks a document.*
MOVE_DOCUMENT	*Moves a document.*
PURGE_DOCUMENT	*Purges a document.* *A purged document is definitively removed from the application.*
RENAME_DOCUMENT	*Renames a document.*
SET_DOCUMENT_PROPERTIES	*Sets document properties.* *For example when a user is subscribed to a document.*
SET_DOCUMENT_NODE_CLASS	*Set the node class.*
UNLOCK_DOCUMENT	*Unlocks a document*
DOWNLOAD_DOCUMENT	*Downloads a document.*
DOWNLOAD_PREVIEW	*Previews a document.*
DOWNLOAD_PRINT	*Prints a document.*
SET_DOCUMENT_ANNOTATIONS	*Set annotations in a document.*
GET_DOCUMENT_ANNOTATIONS	*Get annotations of a document.*
SET_DOCUMENT_DISPOSITION_STEP	*Set the disposition step.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Folder actions

Field / Property	*Description*
COPY_FOLDER	*Copies a folder.*
CREATE_FOLDER	*Creates a folder.*
DELETE_FOLDER	*Deletes a folder.*
GET_CHILDREN_FOLDERS	*Retrieves a folder list in a node.* *List of folders in a folder or record.*
GET_FOLDER_CONTENT_INFO	*Counts nodes into a folder (documents, folders, records and mails that are into).*
GET_FOLDER_PROPERTIES	*Retrieves all folder properties.* *Does not include metadata. it has a direct relationship with panel properties when a folder is selected.*
MOVE_FOLDER	*Moves a folder.*
PURGE_FOLDER	*Purges a folder.* *A purged folder is definitively removed from application.*
RENAME_FOLDER	*Renames a folder.*
SET_FOLDER_STYLE	*Sets folder style.* *Default folder icons can be changed.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

EXPORT_FOLDER_AS_JAR	*Folder contents exported as a JAR file.*

Mail actions

Field / Property	*Description*
CREATE_MAIL	*Creates a mail.*
GET_MAIL_PROPERTIES	*Retrieves all mail properties.* *Does not include metadata. It has a direct relationship with panel properties when a mail is selected.*
DELETE_MAIL	*Deletes a mail.*
PURGE_MAIL	*Purges a mail.* *A purged mail is definitively removed from application.*
RENAME_MAIL	*Renames a mail.*
MOVE_MAIL	*Moves a mail.*
COPY_MAIL	*Copies a mail.*
GET_CHILDREN_MAILS	*Retrieves mail list in some node.* *List of mail in a folder or record.*
DOWNLOAD_MAIL	*Downloads a mail*
SEND_MAIL_LINK	*Send a mail with a node link.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

SEND_MAIL_ATTACHMENT	*Send a mail with document attachments.*
CREATE_MAIL_ATTACHMENT	*Create a mail with document attachments.*
DELETE_MAIL_ATTACHMENT	*Delete a mail with document attachments.*
GET_MAIL_ATTACHMENTS	*Retrieves a mail with attachments.*
SEND_MAIL_LINK	*Send a mail link.*
DOWNLOAD_MAIL	*Download a mail.*
SET_MAIL_NODE_CLASS	*Set node class.*
SET_MAIL_DISPOSITION_STEP	*Set the disposition step.*

Record actions

Field / Property	*Description*
CREATE_RECORD	*Creates a record.*
GET_RECORD_PROPERTIES	*Retrieves all record properties.* *Does not include metadata. It has a direct relationship with panel properties when a record is selected.*
DELETE_RECORD	*Deletes a record.*
PURGE_RECORD	*Purges a record.* *A purged record is definitively removed from application.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

RENAME_RECORD	*Renames a record.*
MOVE_RECORD	*Moves a record.*
COPY_RECORD	*Copies a record.*
GET_CHILDREN_RECORDS	*Retrieves record list in some node.* *List of record in a folder or record.*
SET_RECORD_TITLE	*Sets title.*
SET_RECORD_NODE_CLASS	*Set record node class.*
ADD_RECORD_NODE_CLASS_CHILDREN	*Add node class children.*
SET_RECORD_DISPOSITION_STEP	*Set the disposition step.*

Property group actions

Field / Property	*Description*
ADD_PROPERTY_GROUP	*When add metadata to a node (document, folder, mail or record).*
REMOVE_PROPERTY_GROUP	*When remove metadata to a node (document, folder, mail or record).*
SET_PROPERTY_GROUP_PROPERTIES	*When update metadata to a node (document, folder, mail or record).*
GET_PROPERTY_GROUP_PROPERTIES	*Retrieves all metadata properties of a node (document, folder, mail or record).*
GET_PROPERTY_GROUP_PROPERTY	*Retrieves a single metadata group properties of a node (document, folder, mail or record).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Node actions

Field / Property	Description
SUBSCRIBE_NODE	*When an user subscribe a node (document, folder or record).*
UNSUBSCRIBE_NODE	*When an user desubscribe from a node (document, folder or record).*
NODES_TO_ZIP	*Exports nodes to ZIP.*
RESTORE_NODE_VERSION	*Restore version of a node (document, folder or record).*

Repository actions

Field / Property	Description
PURGE_TRASH	*Trash is purged.* *All related nodes are definitively removed from application.*

Admin actions

These actions are only done by users with Administration grants.

Field / Property	Description
ADMIN_ACTIVITY_LOG	*The administrator uses "activity log".*
ADMIN_ACTIVE_SESSIONS	*The administrator uses Logged users.*
ADMIN_USER_CREATE	*Creates a user.*
ADMIN_USER_EDIT	*Edits a user.*
ADMIN_USER_DELETE	*Deletes a user.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ADMIN_USER_ACTIVE	*Enables or disables a user.*
ADMIN_ROLE_CREATE	*Creates a role.*
ADMIN_ROLE_EDIT	*Edits a role.*
ADMIN_ROLE_DELETE	*Deletes a role.*
ADMIN_ROLE_ACTIVE	*Enables or disables a role.*
ADMIN_CHECK_EMAIL	*Checks email poll.* *The application reads email accounts to import emails.*
ADMIN_CONFIG_CREATE	*Creates a configuration parameters.*
ADMIN_CONFIG_EDIT	*Edits a configuration parameter.*
ADMIN_CONFIG_DELETE	*Deletes a configuration parameter.*
ADMIN_CRONTAB_CREATE	*Creates a crontab task.*
ADMIN_CRONTAB_EDIT	*Edits a contab task.*
ADMIN_CRONTAB_DELETE	*Deletes a crontab task.*
ADMIN_CRONTAB_EXECUTE	*Executes a crontab task.*
ADMIN_DATABASE_QUERY	*Executes a database query.*
ADMIN_DATABASE_UPDATE	*Executes an update database query.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ADMIN_LANGUAGE_CREATE	*Creates a new laguage.*
ADMIN_LANGUAGE_EDIT	*Edits a language.*
ADMIN_LANGUAGE_DELETE	*Deletes a language.*
ADMIN_LANGUAGE_IMPORT	*Imports a language.*
ADMIN_LOGCAT_LIST	*The administrator user logcat tool.*
ADMIN_LOGCAT_VIEW	*The administrator user logcat tool.*
ADMIN_LOGGED_USERS	*The administrator uses Logged users.*
ADMIN_MAIL_ACCOUNT_CREATE	*Creates a mail account.*
ADMIN_MAIL_ACCOUNT_EDIT	*Edits a mail account.*
ADMIN_MAIL_ACCOUNT_DELETE	*Deletes a mail account.*
ADMIN_MAIL_ACCOUNT_CHECK	*Checks a mail account.*
ADMIN_MAIL_FILTER_CREATE	*Creates a mail account filter.*
ADMIN_MAIL_FILTER_EDIT	*Edits a mail account filter.*
ADMIN_MAIL_FILTER_DELETE	*Deletes a mail account filter.*
ADMIN_MAIL_FILTER_RULE_CREATE	*Creates a mail account rule.*
ADMIN_MAIL_FILTER_RULE_EDIT	*Edits a mail account rule.*

ADMIN_MAIL_FILTER_RULE_DELETE	*Delete a mail account rule.*
ADMIN_MIME_TYPE_CREATE	*Creates a new MIME type*
ADMIN_MIME_TYPE_EDIT	*Edits MIME type.*
ADMIN_MIME_TYPE_DELETE	*Deletes MIME type.*
ADMIN_USER_PROFILE_CREATE	*Creates a profile.*
ADMIN_USER_PROFILE_EDIT	*Edits a profile.*
ADMIN_USER_PROFILE_DELETE	*Deletes a profile.*
ADMIN_PROPERTY_GROUP_REGISTER	*Registers a metadata group.*
ADMIN_PROPERTY_GROUP_LIST	*Lists metadata groups.*
ADMIN_REPORT_CREATE	*Creates a report.*
ADMIN_REPORT_EDIT	*Edits a report.*
ADMIN_REPORT_DELETE	*Deletes a report.*
ADMIN_REPORT_EXECUTE	*Executes a report.*
ADMIN_REPOSITORY_SEARCH	*Performes a Lucene search.*
ADMIN_REPOSITORY_REINDEX	*Reindexes Lucene repository.*
ADMIN_REPOSITORY_UNLOCK	*Unlocks a node from a repository view.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ADMIN_REPOSITORY_CHECKIN	*Updates a node from a repository view.*
ADMIN_REPOSITORY_EDIT	*Edits a node from a repository view.*
ADMIN_REPOSITORY_LIST	*Refreshes folder contents from repository view.*
ADMIN_REPOSITORY_REMOVE_CONTENT	*Purges folder lo dentro*
ADMIN_REPOSITORY_REMOVE_CURRENT	*Deletes all folder todo*
ADMIN_WORKFLOW_REGISTER	*Registers a workflow.*
ADMIN_PROCESS_DEFINITION_DELETE	*Deletes a workflow process definition.*
ADMIN_PROCESS_INSTANCE_DELETE	*Deletes a workflow process instance.*
ADMIN_PROCESS_INSTANCE_END	*Workflow process instance finished.*
ADMIN_PROCESS_INSTANCE_RESUME	*Workflow process instance resumed.*
ADMIN_PROCESS_INSTANCE_SUSPEND	*Workflow process instance suspended.*
ADMIN_PROCESS_INSTANCE_ADD_COMMENT	*Adds comment on a workflow process instance.*
ADMIN_PROCESS_INSTANCE_VARIABLE_DELETE	*Deletes variable on a workflow process instance.*
ADMIN_PROCESS_INSTANCE_VARIABLE_ADD	*Adds variable on a workflow process instance.*
ADMIN_TASK_INSTANCE_SET_ACTOR	*Sets actor on a a workflow task instance.*
ADMIN_TASK_INSTANCE_START	*Workflow task instance started.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ADMIN_TASK_INSTANCE_END	*Workflow task instance ended.*
ADMIN_TASK_INSTANCE_SUSPEND	*Workflow task instance suspeded.*
ADMIN_TASK_INSTANCE_ADD_COMMENT	*Adds comment on a workflow task instance.*
ADMIN_TASK_INSTANCE_VARIABLE_DELETE	*Deletes variable on a workflow task instance.*
ADMIN_TASK_INSTANCE_VARIABLE_ADD	*Adds variable on a workflow task instance.*
ADMIN_TASK_INSTANCE_RESUME	*Resumes a workflow task instance.*
ADMIN_TOKEN_SIGNAL	*Workflow task signal sent from administration view.*
ADMIN_STAMP_IMAGE_CREATE	*Creates a stamp image.*
ADMIN_STAMP_IMAGE_EDIT	*Edits a stamp image.*
ADMIN_STAMP_IMAGE_DELETE	*Deletes a stamp image.*
ADMIN_STAMP_IMAGE_ACTIVE	*Enables or disables a stamp image.*
ADMIN_STAMP_TEXT_CREATE	*Creates a stamp text.*
ADMIN_STAMP_TEXT_EDIT	*Edits a stamp text.*
ADMIN_STAMP_TEXT_DELETE	*Deletes a stamp text.*
ADMIN_STAMP_TEXT_ACTIVE	*Enables or disables a stamp text.*
ADMIN_TWITTER_ACCOUNT_CREATE	*Creates a twitter account.*
ADMIN_TWITTER_ACCOUNT_EDIT	*Edits a twitter account.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ADMIN_TWITTER_ACCOUNT_DELETE	*Deletes a twitter account.*
ADMIN_USER_CONFIG_EDIT	*Changes a user profile.*
ADMIN_SCRIPTING	*Executes scripting.*
ADMIN_OCR_TEMPLATE_LIST	*Lists ocr templates.*
ADMIN_OCR_TEMPLATE_DELETE	*Deletes ocr template.*
ADMIN_OCR_TEMPLATE_CREATE	*Creates ocr template.*
ADMIN_OCR_TEMPLATE_EDIT	*Edits ocr template.*
ADMIN_OCR_TEMPLATE_GET_FIELDS_LIST	*Lists ocr template fields.*
ADMIN_OCR_TEMPLATE_FIELD_CREATE	*Creates ocr template field.*
ADMIN_OCR_TEMPLATE_CONTROL_FIELD_CREATE	*Creates ocr template control field.*
ADMIN_OCR_TEMPLATE_FIELD_EDIT	*Edits ocr template field.*
ADMIN_OCR_TEMPLATE_FIELD_DELETE	*Deletes ocr template field.*
ADMIN_OCR_TEMPLATE_CONTROL_FIELD_DELETE	*Deletes ocr template control field.*
ADMIN_OCR_TEMPLATE_CONTROL_FIELD_EDIT	*Edits ocr template control field.*
ADMIN_MESSAGE_QUEUE_CREATE	*Creates a new message.* *For more information see Message queue.*

ADMIN_MESSAGE_QUEUE_UPDATE	*Updates message.*
ADMIN_MESSAGE_QUEUE_DELETE	*Deletes a message.*
ADMIN_DELETE_MESSAGE	*Deletes a message*

Misc actions

Field / Property	*Description*
MISC_OPENKM_START	*Starts application.*
MISC_OPENKM_STOP	*Stops application.*
MISC_STATUS	*Application check.* *Several internal checks to ensure application is alive:* • *Database connection.* • *Repository connection.*
MISC_TEXT_EXTRACTION_FAILURE	*Text extraction failure.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow

In Workflow view are shown the registered workflows. In this view the user can register new workflows and view or modify existing workflow process instances.

Workflow are shown in a table structure.

Process ID	Process Name	Version	Action

Description:

Property	Description
Process ID	*Each workflow has an unique Process ID.*

Process Name	*The workflow Process Name is set into the workflow process definition.*
	The workflow process definition is an XML. In the first tag is set the Process Name.
	`<?xml version="1.0" encoding="UTF-8"?>`
	`<process-definition xmlns="urn:jbpm.org:jpdl-3.2" name="purchase">`
	`...`
	`</process-definition>`

Version	*Workflow version.* *When is a new workflow is registered, if exists a previous workflow with the same Process Name, then the version will be increased in one unit.*
Actions	*Available actions:* • *View (manage process intances, form definition, process image).* • *Delete*

Register workflow

- *On the right bottom,* ***select the .par file*** *(workflow file type).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click* ***Register process definition button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow forms definition

Workflow forms are based on formal XML definition. The DTD (Document Type Definition) defines the structure and the legal elements and attributes of an XML document.

*You can see the latest DTD in our website http://www.openkm.com/dtd/workflow-forms-**{major}.{minor}**.dtd where **{major}** and **{minor}** values are two numbers (major release and minor release). For example the location for version 2.6 of the dtd is http://www.openkm.com/dtd/workflow-forms-2.6.dtd*

Each application version supports only a couple of DTD definitions, read: Application version compatibility table to see which are supported by your application.

Basic XML skeleton

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                               "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>

</workflow-forms>
```

- *The **DOCTYPE** is a formal definition of some DTD. In the previous example the **workflow-forms-2.6.dtd**.*
- *The XML begins with **<workflow-forms> tag** and ends with **</workflow-forms>**. Into this hicheracy are defined the metada groups.*

Workflow task form definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.0//EN"
                               "http://www.openkm.com/dtd/workflow-forms-2.0.dtd">
<workflow-forms>
  <workflow-form task="evaluate price">

  </workflow-form>
</workflow-forms>
```

- *Each **<workflow-form> ... </workflow-form>** block identify the **beginning and end** of new **workflow form task definition**.*
- *Each task has a **unique name**, **"evaluate price"**.*
- *Must not use special characters for name.*

Attribute	*Description*	*Required*
task	*Unique field identifier.*	*true*

	Two metadata groups can't have the same name. ***Name must be unique****.* *The name must start with "**okg:**".* *Must not use special character for task name. We suggest (0-9a-zA-Z_).*	
defaultValueClassName	*Set custom JAVA class for setting initial default values.*	*false*

Two form task can't have the same name.

Must not use special character for task name. We suggest (0-9a-zA-Z_).

Basic example

Available field types:

- *Checkbox*
- *Button*
- *Download*
- *Input*
- *Iframe*
- *Select*
- *Separator*
- *Suggestbox*
- *Text*
- *TextArea*
- *Upload*
- *VerticalPanel*
- *HorizontalPanel*

More information about field type elements at Workflow fields

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.0//EN"
                               "http://www.openkm.com/dtd/workflow-forms-2.0.dtd">
<workflow-forms>
  <workflow-form task="evaluate price">
    <input label="Purchase price" name="price" data="price" readonly="true" />
    <textarea label="Purchase description" name="description" data="description" read
    <button name="approve" label="Approve" transition="approve"/>
    <button name="deny" label="Deny" transition="deny"/>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Using file system DTD

If the server has no access to the Internet, then the DTD file must be accessible in some way, for example can store a copy in your application server.

The XML definition must be changed to refer to the new DTD location, otherwise you will get and application error while trying to process a DTD file.

It can be a file system (example below) or a valid intranet URL.

```
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.6//EN"
                                 "file:///home/openkm/workflow-forms-2.6.dtd">
```

- *In the example, **/home/openkm/workflow-forms-2.6.dtd** is the file system path to the file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow fields

These are the available field types:

Field type	Description
Checkbox	Workflow Checkbox field
Button	Workflow Button field
Download	Workflow Download field
Input	Workflow Input field
Print	Workflow Print field
Select	Workflow Select field
Separator	Workflow Separator field
Suggestbox	Workflow Suggestbox field
Text	Workflow Text field
TextArea	Workflow TextArea field
Upload	Workflow Upload field
VerticalPanel	Workflow VerticalPanel field
HorizontalPanel	Workflow HorizontalPanel field

Some field types can have children elements, validator or option elements. This hierarchy is permitted:

- *workflow form*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Checkbox*
    - *Validator*
- *Button*
- *Download*
    - *Node*
    - *Validator*
- *Input*
    - *Validator*
- *Print*
    - *Node*
    - *Validator*
- *Select*
    - *Option*
    - *Validator*
- *Separator*
- *Suggestbox*
    - *Validator*
- *Text*
- *TextArea*
    - *Validator*
- *Upload*
    - *Validator*
- *VerticalPanel*
    - *Checkbox*
    - *Button*
    - *Donwload*
    - *Input*
    - *Print*
    - *Select*
    - *Separator*
    - *Suggestbox*
    - *Text*
    - *TextArea*
    - *Upload*
    - *VerticalPanel*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *HorizontalPanel*

- *HorizontalPanel*
  - *Checkbox*
  - *Button*
  - *Donwload*
  - *Input*
  - *Print*
  - *Select*
  - *Separator*
  - *Suggestbox*
  - *Text*
  - *TextArea*
  - *Upload*
  - *VerticalPanel*
  - *HorizontalPanel*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Button field

Buttons are typically used by users - in user interface - for decision making.

> *For example two buttons, accept and deny. One for accept something and the other one to deny.*

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*A unique field identifier* *Two fields can't have the same name into the same task definion.* ***The Name must be unique****.*	*true*
transition	*If present, a process instance token will follow the indicated transition.*	*false*
confirmation	*The button will show as a popup with an operation confirmation message. The message text used is confirmation value.*	*false*
validate	*Default value is "true".*	*false*
style	*Icon style applied to the button. Possible values:* • *yes* • *no* • *add* • *delete* • *download* • *downloadZip* • *home* • *view* • *change* • *compact* • *clean* • *search* • *save*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	• *comment*	
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.7//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.7.dtd">
<workflow-forms>
  <workflow-form task="evaluate price">
    <button name="deny" label="Deny" transition="deny"/>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Download field

This workflow is used to download a document.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique name identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
data	*When present,it is an identifier used to load data.* *For example to dynamically load the uuid or path of the node to download.*	*false*

Child elements:

- *Node. For more information see Workflow Node element.*
- *Validator. For more information see Workflow Validator element.*

Basic download example

- *Download button with node element.*
- ***uuid** is a valid document uuid.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dt
<workflow-forms>
  <workflow-form task="download test">
    <download name="download" label="Download document">
      <node label="download" uuid="afb8ac33-912d-4515-e29b-1c01b693809a" />
    </download>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Dynamic download example

- *Download button with node element.*
- ***uuid** is a valid document uuid.*
- ***data** is a valid mapping variable. This **variable** is **used to dynamically set uuid attribute** into node element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="download test">
    <download name="download" label="Download document" data="dynamic">
      <node label="download"/>
    </download>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Validator element

The goal of the Validatior element is to ensure that the user provides necessary and properly formatted information needed to successfully complete an operation.

Validator can only be used in combination with other fields:

- *Workflow Input field*
- *Workflow Suggestbox field*
- *Workflow Upload field*
- *Workflow Download field*
- *Workflow Print field*
- *Workflow Checkbox field*
- *Workflow Text field*
- *Workflow TextArea field*
- *Workflow Select field*

Available validators:

name	*Description*
req	*Required value.*
alpha	*Alphanumeric value.*
dec	*Decimal value.*
num	*Numeric value.*
email	*E-mail value.*
url	*URL value.*
maxlen	*Sets the maximum length allowed.*
minlen	*Sets the minimum length required.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

lt	*Less than.*
gt	*Greater than.*
min	*Minimum value.*
max	*Maximum value.*
regexp	*Regular expression.*
plugin	*Create your own form validator.* *More information at: Creating your own Form Validator plugin.*

Parent:

- *Workflow Input field*
- *Workflow Suggestbox field*
- *Workflow Upload field*
- *Workflow Download field*
- *Workflow Print field*
- *Workflow Checkbox field*
- *Workflow TextArea field*
- *Workflow Select field*

Basic example

- *Value required "**req**".*
- *Value must be decimal "**dec**".*
- *The maximum length of the decimal value is 6 digits "**maxlen**".*

```
<input label="Decimal value" name="decimal">
    <validator type="dec"/>
    <validator type="req"/>
    <validator type="maxlen" parameter="6"/>
</input>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Regular expression

If you want a validation which is not included in these prefefined, you can use a regular expression. For example, imagine you want to restrict an input to lowecase ASCII values:

```
<input label="Lowercase value" name="regExp">
    <validator type="regexp" parameter="[a-z]+" />
</input>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Node element

This option element is only used in combination with:

- *Workflow Download field*

The option value represents a list of distinct values in a Selected field.

Attribute	*Description*	*Required*
label	*The text shown as a label in the user interface.*	*true*
path	*Node path. Usually a document path.*	*false*
uuid	*Node uuid. Usually a document valid uuid.*	*false*

Parent :

- *Workflow Download field*

Basic download example

- *Download button with node element.*
- ***uuid** is a valid document uuid.*

```
<workflow-form task="download test">
  <download name="download" label="Download document">
    <node label="download" uuid="afb8ac33-912d-4515-e29b-1c01b693809a" />
  </download>
</workflow-form>
```

Take it as an example. Normally it wants to assign dynamically a uuid, for it is used data attribute.

```
<download name="download" label="Download document" data="dynamic" />
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Upload field

Upload is used to upload a document.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique name identifier* ⚠ *Two fields can't have the same name in the same task defition.* ***Name must be unique****.*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
data	*When present, is an identifier used to load data.* ℹ *For example dynamically set the folderPath or folderUuid.*	*false*
type	*Values:* • *create (the operation will be create new document)* • *update (the operation will be update existing document)* ℹ *When it is not present, the default value is set to create.*	*false*
folderPath	*The folder path where the document will be stored.*	*false*
folderUuid	*The folder uuid where the document will be stored.*	*false*
documentName	*Forced name of the uploaded document.*	*false*
documentUuid	*The uuid of the document to be updated.*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

To upload a new document, you should set the folderPath or folderUuid, optionally documentName if we want to force the document name.

To update an existing document, you should set documentUuid and put type to update.

Child elements:

- *Validator. For more information see Workflow Validator element.*

Basic upload example

- *Upload a new document to a folder.*
- ***folderUuid** is a valid folder uuid. You can also use **folderPath** and set a path, which is more human readable.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dt
<workflow-forms>
  <workflow-form task="upload test">
    <upload name="upload" label="Upload document" folderUuid="fca2d85e-0e01-418c
  </workflow-form>
</workflow-forms>
```

Document update

- *Create a new version of a given document.*
- ***documentUuid** is the UUID of the document to be replaced.*
- *Note the **type** parameter which is set to update.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                               "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="upload test">
    <upload name="upload" label="Upload document" type="update" documentUuid="e6a0630
  </workflow-form>
</workflow-forms>
```

Upload example with dynamic parameters

- *Upload a document into a non-predefined folder.*
- ***data** is a valid mapping variable. This **variable** is **used to dynamically set folderPath attribute**.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                               "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="upload test">
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    <upload name="upload" label="Upload document" data="updData" />
  </workflow-form>
</workflow-forms>
```

This is the Java code which set the data and should be executed before the task which displays the form element:

```
import org.jbpm.graph.def.ActionHandler;
import org.jbpm.graph.exe.ExecutionContext;

import com.openkm.bean.form.Upload;

public class UploadAction implements ActionHandler {
    private static final long serialVersionUID = 1L;

    @Override
    public void execute(ExecutionContext ctx) throws Exception {
        Upload upd = new Upload();
        upd.setFolderPath("/okm:root/dstFld");
        ctx.setVariable("updData", upd);

        // Go to next node
        ctx.getToken().signal();
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Print field

Deprecated from version 7.1.18

Used to print a document.

Attribute	Description	Required
label	*The text shown as label in user interface.*	*true*
name	*Unique name identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
data	*When present, is an identifier used to load data.* *For example dynamically set the uuid or path of the document to download.*	*false*

Child elements:

- *Node. For more information Workflow Node element.*
- *Validator. For more information Workflow Validator element.*

Basic print example

- *Print a document*
- ***uuid** is a valid document uuid.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dt
<workflow-forms>
  <workflow-form task="print test">
    <print name="print" label="Print document">
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
      <node label="print" uuid="afb8ac33-912d-4515-e29b-1c01b693809a" />
    </print>
  </workflow-form>
</workflow-forms>
```

Dynamic print example

- *Print an existing document*
- ***data** is a mapping valid variable. This **variable** is **used to dynamically set uuid attribute** into the node.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="print test">
    <print name="print" label="Print document" data="dynamic">
      <node label="print">
    </print>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Checkbox field

Checkbox

Best used to represent boolean data:

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
readonly	*Set field, it will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*
data	*When present, is an identifier used to load data.* *For example to dynamically load the true or false value of the checkbox.*	

Child elements:

- *Validator. For more information see Workflow Validator element*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="checkbox test">
    <checkbox label="Checkbox label" name="checkbox" />
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Input field

In mosts cases is used to enter free form text, although can be used to save http links, date data or refer to application folder (as internal link to this folder).

Attribute	*Description*	*Required*
label	*The text shown as a label in the user interface.*	*true*
name	*Unique field identifier.* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
type	*Type value. Allowed types:* • *text.* • *date.* • *link.* • *folder.*	*true*
value	*Can specify a default value for this component.* *There's a special case for date types. Data is stored in format* ***"yyyyMMddHHmmss"****, take care of it when you're using OpenKM API.*	*false*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
data	*When present, is an identifier used to load data.* *For example to dynamically load the Input value.*	
readonly	*Set field will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

timeFormat	*When input type is date we can specify if want to be able to see and modify hours and minutes. Allowed values:* • *hm* • *h* • *none*	*false*

Child elements:

- *Validator. For more information see Workflow Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="input test">
    <input label="Input label" name="input" />
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Select field

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier.* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
type	*Type value. Allowed types:* • *simple (single choice allowed).* • *multiple (multiple choice allowed).*	*true*
table	*Metadata table name.* *Application can store values into metadata table, that can be used in this query.*	*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
optionsQuery	*A metadata query or SQL query used to get the select options.*	*
className	*A class used to get the select options.*	*
data	*When present, is an identifier used to load data.*	*false*
optionsData	*When present, is an identifier used to load options.*	*false*
readonly	*Set field will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Child elements:

- *Option. Couples of XML* ***<option label="label text" value="unique identifier value"/>*** *values or* ***<option label="label text" value="unique identifier value" parentValue="parent unique identifier value"/>*** *in case is set some hierarchical relationship.*
- *Validator. For more information see Workflow Validator element.*

***** There're three options to get option values:*

- *Directly defined in XML.*
- *Metadata table (table + optionsQuery).*
- *className.*

One of these options must be defined, otherwise you will get an error. A combination of several of these cannot be done, only one can be choosen.

Basic select example

- *Type is equal to* ***simple****.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="select test">
    <select label="select label" name="select" type="simple">
      <option label="one" value="001" />
      <option label="two" value="002" />
      <option label="three" value="003" />
    </select>
  </workflow-form>
</workflow-forms>
```

Allow multiple choice

- *Type is equal to* ***multiple****.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="select test">
    <select label="select label"  name="select" type="multiple" >
      <option label="one" value="001"  />
      <option label="two" value="002"  />
      <option label="three" value="003"  />
    </select>
  </workflow-form>
```

```
</workflow-forms>
```

Hierarchical relationship

- *The field option value of "**childrenSelect**" has a parent relation with "**parentSelect**". It's defined by **parentElement="parentSelect"**.*
- *The option value "**benefits**" of the field "**childrenSelect**" has a parent value "**directives**" from field "**parentSelect**".*

Tree relationships:

- *directives*
  - *benefits*
  - *eligibility*
  - *services*
- *handbook*
  - *basic*
  - *normal*
  - *advanced*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="select test">
    <select label="Parent" name="parentSelect" type="simple">
      <option value="directives" label="Directives" />
      <option value="handbook" label="Handbook Paragraphs" />
    </select>
    <select label="Children" name="childrenSelect" type="simple" parentElement="paren
      <option value="benefits" label="Benefits" parentValue="directives" />
      <option value="elegibility" label="Eligibility" parentValue="directives" />
      <option value="services" label="Services" parentValue="directives" />
      <option value="basic" label="Basic" parentValue="handbook" />
      <option value="normal" label="Advanced" parentValue="handbook" />
      <option value="advanced" label="Normal" parentValue="handbook" />
    </select>
  </workflow-form>
</workflow-forms>
```

Based on SQL query

*We can also get these option values using a SQL query. This SQL sentence should be put on the **optionsQuery** attribute, but the **table** attribute should be empty (otherwise this query is used as a metadata query). Note that the first element returned in every row is the option value and the second one is the option label.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<workflow-forms>
  <workflow-form task="select test">
    <select label="country" name="country" type="simple"
            optionsQuery="select CT_ID, CT_NAME from COUNTRY order by CT_NAME"/>
  </workflow-form>
</workflow-forms>
```

The table used in this sample is defined as:

```
create table COUNTRY (
  CT_ID varchar(2),
  CT_NAME varchar(32),
  primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Based on className

To help getting a list from external applications or other purposes you can build your own class to retrieve from these systems the option values.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="select test">
    <select label="select label" name="okp:consulting.select" type="simple"
            className="com.openkm.plugin.form.values.OptionSelectUserList" />
  </workflow-form>
</workflow-forms>
```

For more information read: Creating your own Option Select Values plugin.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Separator field

Draw a separator between fields.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
readonly	*Set field, it will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="separator test">
    <input label="Input label" name="input1" />
    <separator label="Separator label" name="separator" />
    <input label="Input label" name="input2" />
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Suggestbox field

This is an implementation of the typical AJAX suggest box component:

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique field identifier.* ⚠ *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
table	*Metadata table name.* *Application can store values in a metadata table, that can be used in this query.*	*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
filterQuery	*A Hibernate query to filter the data. Use {0} to place the user input.*	*
valueQuery	*A Hibernate query to obtain the value of an identifier. Use {0} to place the identifier.*	*
className	*A class used to get the selected options.*	*
filterMinLen	*The minimum chars in input to begin to filter results.*	*true*
dialogTitle	*The title of the HTML dialog.*	*true*
data	*When present, is an identifier used to load data.*	*false*
readonly	*Set field, it will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Child elements:

- *Validator. For more information see Workflow Validator element.*

** There're three options to get suggestbox values (**use one of them is mandatory**):*

- *filterQuery + valueQuery*
- *className*

No other combination of them is allowed.

Based on filterQuery and valueQuery (without table attribute)

*Now **filterQuery** and **valueQuery** could be a regular SQL query.*

You should only use {0} as the parameter entered by the user to filter the values from the table.

Create COUNTRY table:

```
create table COUNTRY (
  CT_ID varchar(2),
  CT_NAME varchar(32),
  primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Metadata definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.5//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.5.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
  <suggestbox label="country" name="okp:consulting.suggestbox"
     width="200px" dialogTitle="Choose country" filterMinLen="3"
    filterQuery="select ct_id, ct_name from country where ct_name like '%{0}%' order
    valueQuery="select ct_id, ct_name from country where ct_id='{0}'" />
  </property-group>
</property-groups>
```

Based on className

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

To help getting a list from external applications or other purposes, your own class can be built to retrieve the option values from those systems .

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.5//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.5.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
  <suggestbox label="country" name="okp:consulting.suggestbox"
     width="200px" dialogTitle="Choose country" filterMinLen="3"
    className="com.openkm.plugin.form.values.SuggestBoxUserList" />
  </property-group>
</property-groups>
```

For more information read Creating your own Suggestbox plugin

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Text field

Print a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
data	*When present, is an identifier used to load data.*	*false*
readonly	*Set field, it will not be able to be modified in user interface. By default the value is "* ***false****".*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="text test">
    <text label="text label" name="consulting_text" width="200px" />
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow TextArea field

Print a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as a label in user interface.*	*true.*
name	*Unique field identifier* ⚠ *Two fields can't have the same name in the same task defition.* ***Name must be unique****.*	*true.*
value	*Default value for the component.*	*false.*
width	*The width of the HTML element.*	*false.*
height	*The height of the HTML element.*	*false.*
data	*When present, is an identifier used to load data.*	*false.*
readonly	*Set field, it will not be able to be modified in user interface. By default the value is "* ***false****".*	*false.*

Child elements:

- *Validator. For more information see: Workflow Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="textarea test">
    <textarea label="textarea label" name="textarea" width="200px"/>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow Option element

The option element is only used in combination with Workflow Select field. The option value represents a list of distinct values to select.

Attribute	Description	Required
label	*The text shown as label in the user interface.*	*true*
value	*Unique value identifier.* ⚠ *Two options can't have the same value.* ***Value must be unique***.	*true.*
parentValue	*When present, it specifies the option parent.* ⚠ *The parent must be present in other select.*	*false*
selected	*When present, it specifies that an option should be pre-selected when the form is rendered.*	*false*
cssClass	*CSS class.* ℹ *More information about creating your own styles at: CSS.*	*false*

Parent:

- *Select. For more information, Please see: Workflow Select field.*

Basic select with selected option

- *attribute selected=true, when present, it specifies that an option should be pre-selected when the page loads.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="textarea test">
    <select label="select label"  name="test" type="simple">
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
        <option label="one" value="001" />
        <option label="two" value="002" selected="true"/>
        <option label="three" value="003" />
    </select>
  </workflow-form>
</workflow-forms>
```

Hierarchical relationship

- *The field option value of "**children**" have a parent relation with "**parent**". It's defined by **parentElement="parent"**.*
- *The option value "**benefits**" of the field "**children**" has a parent value "**directives**" from field "**parent**".*

Tree relationships:

- *directives*
  - *benefits*
  - *eligibility*
  - *services*
- *handbook*
  - *basic*
  - *normal*
  - *advanced*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="textarea test">
    <select label="Provincia" name="parent" type="simple">
        <option value="directives" label="Directives" />
        <option value="handbook" label="Handbook Paragraphs" />
    </select>
    <select label="Provincia" name="children" type="simple" parentElement="parent">
        <option value="benefits" label="Benefits" parentValue="directives" />
        <option value="elegibility" label="Eligibility" parentValue="directives"/>
        <option value="services" label="Services" parentValue="directives"/>
        <option value="basic" label="Basic" parentValue="handbook" />
        <option value="normal" label="Advanced" parentValue="handbook"/>
        <option value="advanced" label="Normal" parentValue="handbook"/>
    </select>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow VerticalPanel field

Vertical layout that allow you to sort components inside.

Attribute	*Description*	*Required*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="verticalpanel test">
    <vpanel name="vertical1" />
       <input label="Input label" name="input1" />
    </vpanel>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Workflow HorizontalPanel field

Horizontal layout that allow you to sort components inside.

Attribute	*Description*	*Required*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Workflow Forms 2.6//EN"
                                "http://www.openkm.com/dtd/workflow-forms-2.6.dtd">
<workflow-forms>
  <workflow-form task="horizontalpanel test">
    <hpanel name="horizontal1" />
       <input label="Input label" name="input1" />
    </hpanel>
  </workflow-form>
</workflow-forms>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Automation

Automation is a set of controls that perform actions based on a set of rules.

Preliminar concepts.

Each rule is composed of:

Field	Description
Order	*Indicates the order of the evaluation.* *The rules with less*
Name	*Rule description.*
Events	*Available events:* • *Document create.* • *Document update.* • *Document delete.* • *Document rename.* • *Document purge.* • *Document move.* • *Document version restore.* • *Document download from UI.* • *Document download from UI for preview.* • *Document print.* • *Document stamp.* • *Document virus detected* • *Document set node class* • *Folder create.* • *Folder delete.* • *Folder move.* • *Task update* • *Task delete.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Exports folder as JAR.*
- *Mail creation.*
- *Mail delete.*
- *Mail move.*
- *Mail download from UI.*
- *Mail set node class.*
- *Mail import attachments.*
- *Record create.*
- *Record delete.*
- *Record move.*
- *Record set node class.*
- *Add metadata group.*
- *Set metadata group.*
- *Remove metadata group.*
- *Text extraction.*
- *Exports several nodes as ZIP.*
- *Task create.*
- *User create*
- *User login.*
- *User logout.*
- *Note create.*
- *Note update.*
- *Note delete.*

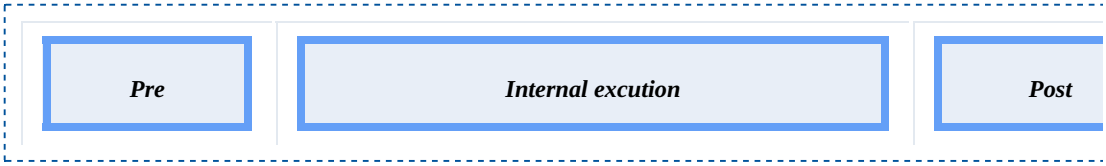
At

The evaluation time.

Available values:

- *Pre.*
- *Post.*

Understanding stages on automation execution:



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	Pre indicate rule execution before an internal event is executed. *Post indicate rule execution after the internal event is executed.* *For example a rules based on "Document creation" event.* • *On "pre" stage the rule is triggered when the document has not yet been created.* • *On "post" stage the rules is triggered after the document has been created*
Exclusive	*When enabled, indicates no order rules should be executed.*
Active	*Enables or disables the rule.*
Validators	*List of validators. Each validator is a condition that must be complied.* *Validations are also ordered. It's a good practice to first set positions that will have more possibiliti to comply.* *When empty, always comply.*
Actions	*List of actions. An action is something that will be done with the node that caused the execution of the rule.* *Nodes types:* • *Document* • *Folder* • *Mail* • *Record* *Example of an action, can be renaming a document immediately after creation.*

Understanding Validators and Actions

The application provides a complete list of available validators and actions from Plugins view. Application plugins' architecture helps administrators to add new features to an existing application.

By default application comes with built in plugins:

- *Built-in Automation Validator plugins*.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Built-in Automation Action plugins*.

Also you can create your own plugins:

- *Creating your own Automation Validation*.
- *Creating your own Automation Action*.

> *More information about how to Register a new plugin.*

Creating a rule

> *Rules are composed of:*
>
> - *General fields.*
> - *Actions and validators definions.*
>
> *A rule creation has two steps. First define the general fields, then the actions and validations definitions.*

- *On top right click on* ***Create new rule***.
- ***Fill in the fields***.
- *Click on* ***Create button***. *(the application will return to the initial screen).*
- *Click on* ***Definition icon***.
- *Add actions and validations according to convenience.*
  - *Click on* ***Add validation icon***.
  - *Click on* ***Add action icon***.
- *When finished, at the top left click on* ***Automation rules button***. *(the application will return to the initial screen).*

Editing a rule

Edit general parameters

- *Click on* ***Edit rule***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Edit action and validation definitions

- *Click on* ***Definition icon***.
- *Modifiy actions and validations according to convenience.*
  - *Validations*
    - *Click on* ***Add validation icon***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on* ***Edit validation icon***.
- *Click on* ***Delete validation icon***.

- *Actions*
  - *Click on* ***Add action icon***.
  - *Click on* ***Edit action icon***.
  - *Click on* ***Delete action icon***.

Deleting a rule

- *Click on* ***Delete rule***.
- *Click on* ***Delete button***.

Built-in Automation Validator plugins

Name	*Apply at*	*Parameters*	*Description*
ContainsFolder	*Post*	*Folder name*	*Valid when the node path contains a folder name.* *The value of the parameter folder name is **test**.* *Node with path **/okm:root/folder1** triggered the event..* *the folder **/okm:root/folder1/test** exists ,then the validation return is **valid**.* *The value of the parameter folder name is **test**.* *Node with path **/okm:root/folder1** triggered the event..* *A folder **/okm:root/folder1/test** doesn't exist, then the validation return is not valid.*
HasCategory	*Post*		
HasKeyword	*Post*	*Keyword name*	*Valid when the node path has a keyword with the same name assigned.*
HasPropertyGroup	*Post*	*Metadata group name*	*Valid when the node path has a metadata group with the same name assigned.*
HasPropertyGroupValue	*Post*	*Metadata field name* *Metadata field value*	*Valid when the node path has a metadata group field with the same value assigned.*
IsOCRDataCaptureFile	*Post*		*Valid when the document is supported by OCR.* *Supported MIME type are:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

			• *PNG* • *GIF* • *JPEG* • *BMP* • *TIFF* • *PDF*
NameContains	*Post*	*Name*	*Valid when the node name contains the name value into.* *The value of the parameter name is **test**.* *Node with path **/okm:root/folder/sometest** triggered the event.* *Folder **sometest** contains the value **test**, then the validation return is **valid**.* *The value of the parameter name is **test**.* *Node with path **/okm:root/folder/somefolder** triggered the event.* *Folder **somefolder** does not contains the value **test**, then the validation return is **not valid**.*
PathContains	*Post*	*Path*	*Valid when the node path starts with a path.* *The value of the parameter path is **/okm:root/folder1**.* *Node with path **/okm:root/folder1/test/temporary** triggered the event.* ***/okm:root/folder1/test/temporary** starts with path **/okm:root/folder1**, then the validation return is **valid**.* *The value of the parameter path is **/okm:root/folder1**.* *Node with path **/okm:root/test/temporary** triggered the event.* ***/okm:root/test/temporary** does not start with path **/okm:root/folder1**, then the validation return*

			*is **not valid**.*
PathContainsFolderName	*Pre,* *Post*	*Folder name*	*Valid when the node path contains a folder name.* *The value of the parameter folder name is **folder1**.* *Node with path **/okm:root/folder1/test/temporary** triggered the event.* ***/okm:root/folder1/test/temporary** contains a folder named **folder1**, then the validation return is **valid**.* *The value of the parameter folder name is **folder1**.* *Node with path **/okm:root/test/temporary/folder1** triggered the event.* ***/okm:root/test/temporary/folder1** contains a folder named **folder1**, then the validation return is **valid**.* *The value of the parameter folder name is **folder1**.* *Node with path **/okm:root/test/temporary** triggered the event.* ***/okm:root/test/temporary** does not contains a folder named **folder1**, then the validation return is **not valid**.*

Built-in Automation Action plugins

Name	Apply at	Parameter	Description
AddCategory	Post	Category	Adds a category to the node that triggered the event.
AddCategoryToWizard	Post		Shows a category popup as part of the process of uploading a new document. ⚠ Only takes effect when an user uploads a new document from the user interface.
AddDigitalSignatureToWizard	Post		Shows a digital signature popup as part of the process uploading a new document.
AddKeyword	Post	Keyword name	Adds a keyword to the node that triggered an event.
AddKeywordToWizard	Post		Shows a keyword popup as part of the process of uploading a new document. ⚠ Only takes effect when an user uploads a new document from user interface.
AddOCRDataCaptureToWizard	Post		Shows an OCR data capture popup as part of the proc of uploading a new document. ⚠ Only take effect when an user uploads a new document from the user interface.
AddPropertyGroup	Post	Metadata	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

		group name	*Add Metadata group.*
AddPropertyGroupToWizard	*Post*	*Metadata group name*	*Shows a Metadata group popup as part of the process uploading a new document.* *Only takes effect when an user uploads a new document from the user interface.*
AddWorkflowToWizard	*Post*	*Workflow name*	*Starts a workflow as part of the process of uploading a new document.* *A popup will be shown if a workflow requires an user to sets parameters for starting the workflow. Otherwise, it will b transparent to the end user.* *Only takes effect when an user uploads a new document from user interface.*
CatalogByMetadata	*Post*		*Used by the fileplan to automatically catalog documer the repository.*
CheckUniqueDocumentName	*Pre*		*In case exist a document with the same name in the wh repository it raises an exception.*
CompactDocumentHistory	*Post*		*Compact the document history removing all older versions.*
ConvertToPDF	*Post*		*Convert a document to PDF.*
CreateDestinationPathByCatalogManager	*Post*		*Used by the fileplan to automatically create the destination path of a node.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

DelCadReferences	*Pre*		*Delete the OpenKM document references of an Autocc document.*
DeleteNode	*Post*		*Deletes a node what is moved into user trash node.*
DownloadAsDocumentVersion	*Post*		*When downloading a document, rename the document including an end the version number.*
DownloadControl	*Post*		*Registers the activity log the triggered a download.* *Available download causes:* • *Folder or document download as ZIP.* • *Document download from user interface.* • *Document download from user interface to preview.* • *Document download from user interface to pr* • *Folder exported as JAR.* • *Email download from user interface.*
ExecuteScripting	*Pre* *Post*	*Code*	*Executes an script.*
ForceGrantActualUser	*Post*	*Permission*	*Grants permission to the user who caused the event in node that triggered the event.* *The super user makes the security change*
ForceRevokeActualUser	*Post*	*Permission*	*Revoke permission to the user who caused the event in node that triggered the event.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

			The super user makes the security change
GrantRole	*Post*	*Role name* *Permission*	*Grants a role permission the node that triggered the e*
GrantUser	*Post*	*User name* *Permission*	*Grants a user permission the node that triggered the event.*
MoveToAZFolder	*Post*	*Folder destination* *Folder deep;*	*Create a folder hierarchy based in the name of the noc and deep and moved the node into.* *Document named "Testing.pdf"* *Folder destination = /okm:root/move/into* *With deep = 3* *The document will be moved at /okm:root/move/into/T/E/S/Testing.pdf*
MoveToFolder	*Post*	*Folder destination*	*Move the node to a new location.*
MoveToYearMonthDayFolder	*Post*	*Folder destination*	*Move the node to a new location with a subfolder structure year / month / day* *Document named "Testing.pdf"* *Folder destination = /okm:root/move/into* *Current date = 2020-05-08* *The document will be moved at /okm:root/move/into/2020/05/08/Testing.p*
NewCadReferences	*Post*		*Create the OpenKM document references of an Autoc document.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OCRDataCapture	*Post*		*Captures data from document.*
OMRDataCapture	*Post*		*Captures data from document.*
PurgeNode	*Post*		*Purge a node. The node is removed from repository.*
RevokeAllRoles	*Post*		*Revokes all roles granted to the node that triggered th event.*
RevokeAllUsers	*Post*		*Revoke all user granted to the node that triggered the event.*
RemoveCategory	*Post*	*Category*	*Removes the category from the node that triggered the event.*
		Keyword	*Remove the keyword from the node that triggered the*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

RemoveKeyword	*Post*	*name*	*event.*
RenameUniqueDocument	*Pre*		*Ensures new documents will have unique name.*
RevokeRole	*Post*	*Role name*	*Revokes the role granted to the node that triggered the event.*
RevokeUser	*Post*	*User name*	*Revokes the user granted to the node that triggered th event.*
Send Mail	*Post*	*Role names* *User names* *Message*	*Send mail to the selected users and roles.* *The message format is the same used in notification messages. For more information take a look at Notification configuration parameters.*
SetInQuarantine	*Post*		*When the antivirus engine detects a virus into a docun moves the document into the quarentine folder /okm:root/quarantine*
SetLanguage	*Post*	*Language*	*Sets the language to the document node that triggered event.* *Language value is based on ISO 639-1.* *For example for Spanish the value is "es" for English "en", etc.*
SetRelationsFromXRef	*Post*		*Set the relations from xref.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

SetSummarization	*Post*		*Set the document summarization.* *Must have enabled the KEA Summarizatio service.*
SetTextExtracted	*Post*	*String*	*Replaces the original content text extracted from the document node that triggered the event.*
StampDocument	*Post*	*Stamp id* *Stamp type*	*Stamp a document.* *Stamp id must be and existing stamp id definition.* *Stamp type values:* • *"text" for text stamping.* • *"images" for images stamping.*
VidSignerDocumentSigned	*Post*		*Mark a document as signed by VidSigner.* *Must have a VidSigner account.*
VidSignerDocumentToSign	*Post*		*Sent a document to sign by VidSigner.* *Must have a VidSigner account.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Stamp

Stamp feature helps you to insert an image into a document.

*This option is hidden when **Stamp** extension is not enabled. Read about it at Enable extensions .*

More information at Stamp extension.

You can set two types of stamp:

- *Based on images.*
- *Based on text.*

*On top right you can switch between "**Text stamp list**" or "**Image stamp list**" with :*

- ***Text stamp list button**.*
- ***Image stamp list button**.*

Stamp are shown in a table structure.

Name	Descripcion	Text	Script	Colour	Active

Create new stamp based on text

When use stamp based on text, the text is converted to an image and then stamped into a document.

Fields description:

Field	Description
Name	*Stamp name.*
Description	*Stamp description.*
Text	*The text to be shown in stamp.*
Script	*When is set the result of the script will be stamped text.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

List of variables what can be used directly:

Variable	*Description*
document	*Document object of the document node.* *For more information see com.openkm.bean.Document class at API description.*

Sample:

```
import java.util;

String value = "";
value = document.getActualVersion().getName();
value += "-" + String.valueOf(new Date());

return value;
```

Layer

Stamped image position.

Available values:

- *Under content.*
- *Over content.*

Opacity

The text opacity. The range of values allowed are from 0 to 1.

*An **Opacity** equals to 0 means a transparent text or image and you won't see it. Set to 0.5 to see an semi-transparent text or set to 1 to be opaque.*

Also play with layer to stamp over or below the default text layer.

Size

Text size.

Align

Text align.

Available values:

- *Left.*

	• *Center.* • *Right.*
Rotation	*Text rotation. The range of values allowed are from 0 to 360.*
Colour	*The text colour.*
Expr. X	*Expression to set the position in X coordinates.*
Expr. Y	*Expression to set the position in Y coordinates.*
Active	*When enabled the stamp will be available to the users.*
Users	*List of users who can use the stamp.*

In Expr. X and Expr. Y input fields you can put more than a simple number. Currently the following macros are defined:

- *PAGE_WIDTH*
- *PAGE_HEIGHT*
- *PAGE_CENTER*
- *PAGE_MIDDLE*

So to center a stamp in the page you can use:

Expr. X	`PAGE_CENTER`
Expr. Y	`PAGE_MIDDLE`

- *Click on* ***Create new Stamp icon***.
- ***Fill the form***.
- *Click on* ***Create button***.

Create new stamp based on image

Fields description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Name</td><td>Stamp name.</td></tr>
<tr><td>Description</td><td>Stamp description.</td></tr>
<tr><td>Image</td><td>The image to be added into the document.</td></tr>
<tr><td>Script</td><td>When is set the result of the script will be stamped text.

List of variables what can be used directly:

<table>
<tr><th>Variable</th><th>Description</th></tr>
<tr><td>document</td><td>Document object of the document node.

For more information see com.openkm.bean.Document class at API description.</td></tr>
</table>

Sample:

<pre>
import java.util;

String value = "";
value = document.getActualVersion().getName();
value += "-" + String.valueOf(new Date());

return value;
</pre>
</td></tr>
<tr><td>Layer</td><td>Stamped image position.

Available values:

• Under content.
• Over content.</td></tr>
<tr><td>Opacity</td><td>The text opacity. The range of values allowed are from 0 to 1.</td></tr>
<tr><td>Expr. X</td><td>Expression to set the position in X coordinates.</td></tr>
</table>

Expr. Y	*Expression to set the position in Y coordinates.*
Active	*When enabled the stamp will be available to the users.*
Users	*List of users who can use the stamp.*

In Expr. X and Expr. Y input fields you can put more than a simple number. Currently the following macros are defined:

- *IMAGE_WIDTH*
- *IMAGE_HEIGHT*
- *PAGE_WIDTH*
- *PAGE_HEIGHT*
- *PAGE_CENTER*
- *PAGE_MIDDLE*

So to center a stamp in the page you can use:

Expr. X	`PAGE_CENTER - IMAGE_WIDTH / 2`
Expr. Y	`PAGE_MIDDLE - IMAGE_HEIGHT / 2`

- *Click on* ***Create new Stamp icon***.
- ***Fill the form***.
- *Click on* ***Create button***.

Edit stamp

- *Click on* ***Edit icon***.
- ***Modify the form***.
- *Click on* ***Edit button***.

Delete stamp

- *Click on* ***Delete icon***.
- *Click on* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Test stamp

- *Click on* *icon.*

Staps what use scripting with variable document can not be tested from administration.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Crontab

The application implements a time-based job scheduler named Crontab. In crontab view, there is a set of jobs that are executed periodically at fixed times or intervals.

Typically automated application maintenance or administration is crontab jobs.

Crontab is shown in a table structure.

Crontab parameters

Field	Description	Required
Name	*Job name.*	*true*
Expression	*Sets when the job is executed.* *Takes into consideration the status of the application when the task will be executed.* *For example, if you stop the application at 00:00 to perform a backup, it delays the task until you consider the backup process will be completed.*	*true*
Class name	*More information at Creating your own Crontab plugin.*	*true*
Mail	*When present, a mail is sent as part of the task once the execution has been completed. You can use several mail addresses using a comma "," as a separator.*	*false*
Last begin	*Shows the last time a process started.*	-
Last end	*Shows the last time a process was completed.*	-
Active	*Enables or disables the process execution.*	*true*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Understanding expression syntax

> *More information https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/scheduling/support/CronSequenceGenerator.htm*

The commands are executed by cron when the minute, hour, and month fields match the current time, and when at least one of the two day fields (day of month, or day of week) match the current time. The scheduler examines crontab entries once every minute. The time and date fields are:

```
* * * * * * command to execute
 â”, â”, â”, â”, â”, â”,
 â”, â”, â”, â”, â”, â”,
 â”, â”, â”, â”, â”, â””â”€â”€â”€â”€â”€â”€â”€ day of week (MON, TUE, WED, THU, FRI, SAT, SUN)
 â”, â”, â”, â”, â””â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€ month (1 - 12)
 â”, â”, â”, â””â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€ day of month (1 - 31)
 â”, â”, â””â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€ hour (0 - 23)
 â”, â””â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€ min
 â””â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€â”€
```

Field	*Values*
seconds	*0-59*
minute	*0-59*
hour	*0-23*
day of month	*1-31*
month	*1-12*
day of week	*MON, TUE, WED, THU, FRI, SAT, SUN*

Samples

*0 0 * * * *" = the top of every hour of every day.*

"/10 * * * * *" = every ten seconds.*

*"0 0 8-10 * * *" = 8, 9 and 10 o'clock of every day.*

*"0 0 6,19 * * *" = 6:00 AM and 7:00 PM every day.*

*"0 0/30 8-10 * * *" = 8:00, 8:30, 9:00, 9:30, 10:00 and 10:30 every day.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*"0 0 9-17 * * MON-FRI" = on the hour nine-to-five weekdays*

"0 0 0 25 12 ?" = every Christmas Day at midnight

Create a new crontab job

> *Crontab job is based in architecture.*
>
> *More information at Creating your own Crontab plugin.*

Edit a crontab job

- *Click on* ***Edit crontab job icon***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Execute a crontab job

- *Click on* ***Execute crontab job icon***

Delete a crontab job

> *Crontab job is based in plugin architecture deployed into $TOMCAT_HOME/plugins. Must delete the jar file in the plugin folder to remove the associated crontab job.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Built in crontab tasks

Task	Default time	Execution	Description
Dashboard Activity Purge	*0 0 0 * * ?*	*Daily execution at 00:00*	*Clean older than one-month dashboard activity log.*
User Mail Importer	*0 */30 * * * ?*	*Execution every 30 minutes.*	*Import mails from registered user mail accounts.*
Text Extractor Worker	*0 */5 * * * ?*	*Execution every 5 minutes.*	*Process text extraction queue.* *For more information Statistics.*
Antivirus Checker Worker	*0 */5 * * * ?*	*Execution every 5 minutes.*	*Process antivirus check queue.* *For more information Statistics.*
Process Pending Tasks	*0 */5 * * * ?*	*Execution every 5 minutes.*	*Execute the pending task.* *For more information Statistics.*
Send Grouped Notifications	*0 0 0 * * ?*	*Daily execution at 00:00*	*Send daily mail notification grouped on single mail.* *For more information Profile Notification tab.*
Purge Old Trash Nodes	*0 0 0 * * ?*	*Daily execution at 00:00*	*Purge old nodes in the trash.*
Check disk space	*0 0 0 * * ?*	*Daily execution at 00:00*	*When there's few free disk space sends an email warning to the administrator.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

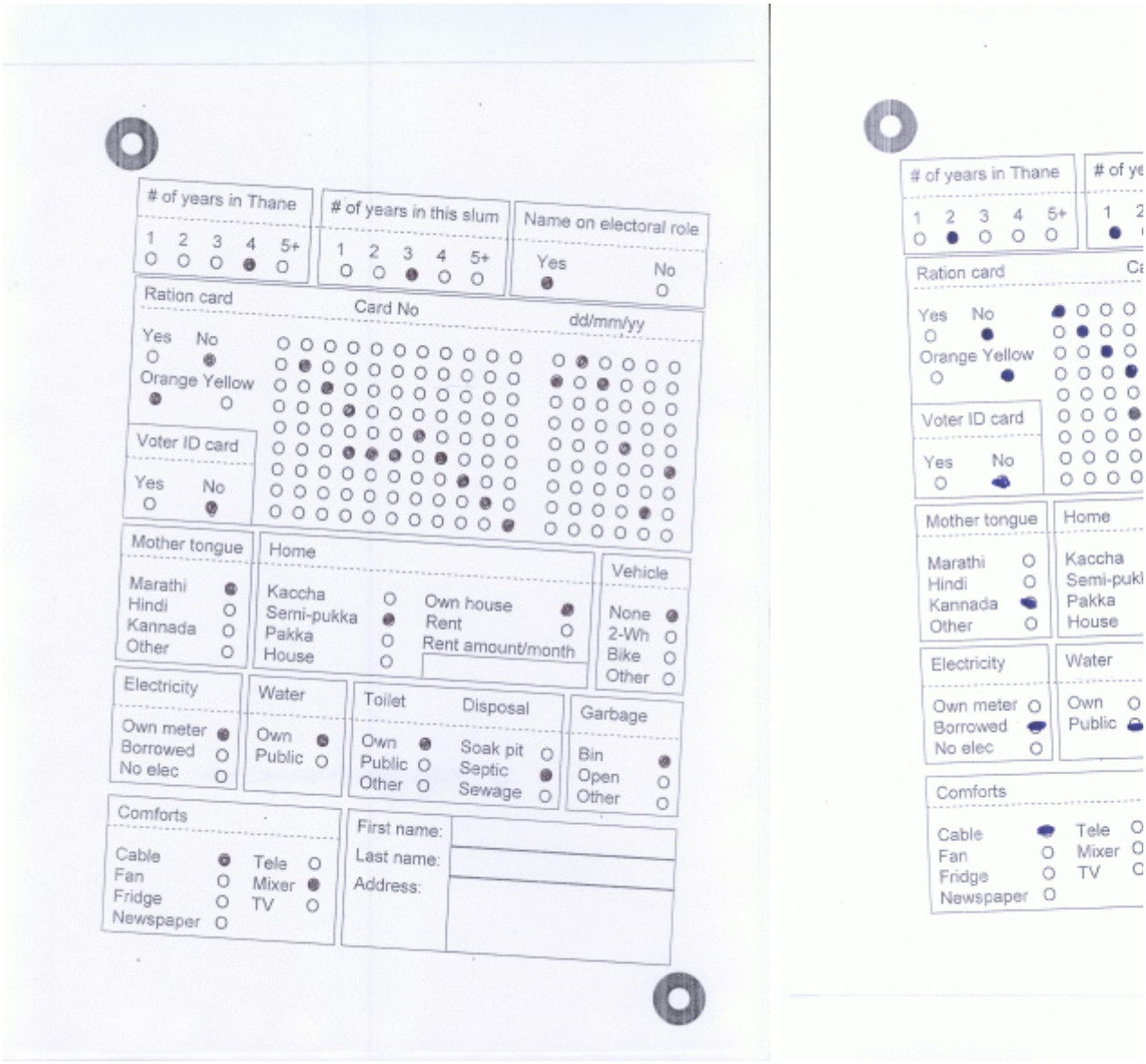
Check Repository Integrity	*0 0 0 * * SAT*	*Execution each Saturday at 00:00*	*Check repository integrity and makes a report.*
Purge Download Preview Tokens	*0 */5 * * * ***	*Execution every 5 minutes.*	*Delete expired preview tokens what have not been consumed.*
Purge Empty Datastore Directories	*0 0 0 * * **	*Daily execution at 00:00*	*Delete the empty folder in the datastore.*
Purge Empty Extraction Directories	*0 0 0 * * **	*Daily execution at 00:00*	*Delete the empty folder in the extraction.*
Purge Empty OCR Template Directories	*0 0 0 * * **	*Daily execution at 00:00*	*Delete the empty OCR template folders.*
Purge Temporal Printing Files	*0 0 0 * * **	*Daily execution at 00:00*	*Delete temporary printing files.*
Purge Tomcat Logs	*0 0 0 * * **	*Daily execution at 00:00*	*Delete old tomcat logs.*
Repository Info	*0 0 0 * * **	*Daily execution at 00:00*	*Update repository stats.*
Text Extractor Issues	*0 0 0 * * **	*Daily execution at 00:00*	*Collect text extraction issues and send an email warning to the administrator.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OMR templates

Application integrates OMR - Optical Mark Recognition - that has been written by Aaditeshwar Seth from the Udai Waterloo chapter, at the University of Waterloo. Udai is a volunteer student organization at the University of Waterloo in Canada and the University of California at San Diego. For more information about it see [OMRProj website](#).

OMR (Optical Mark Recognition) engines is especially suited for extracting data from forms that have been photo-copied and then scanned; hence, the resultant images are likely to suffer from rotations, smudge marks, and random lines here and there. Examples of a few forms that the tool is able to handle with 100% accuracy are shown here.

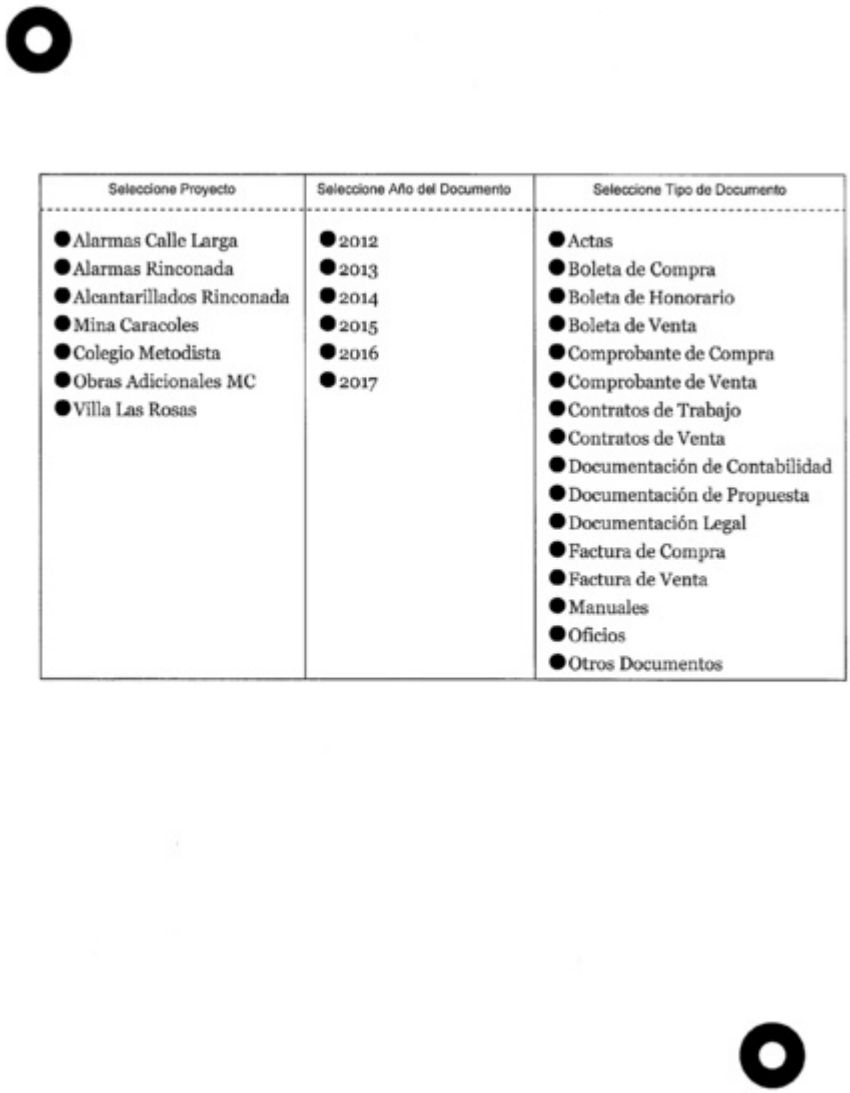


Add a new OMR template

Get a full example Omr_example.zip

Preliminars

- *First make a form in PowerPoint using the template Omr_patterns.zip. Make sure that you do not move the two concentric circles in the top-left and bottom-right corners of the template. These are used by the software to determine the angle of rotation of the form.*
- *Take a printout and make as many photocopies of the form as you want.*
- *On the original printout, color all the marks with a black pen, and scan this marked up printout. Please be as careful as you can to align the printout so that there is no rotation. Some amount of error is fine, but not too much!* ***This scanned image will be the template****.*



Seleccione Proyecto	Seleccione Año del Documento	Seleccione Tipo de Documento
● Alarmas Calle Larga	● 2012	● Actas
● Alarmas Rinconada	● 2013	● Boleta de Compra
● Alcantarillados Rinconada	● 2014	● Boleta de Honorario
● Mina Caracoles	● 2015	● Boleta de Venta
● Colegio Metodista	● 2016	● Comprobante de Compra
● Obras Adicionales MC	● 2017	● Comprobante de Venta
● Villa Las Rosas		● Contratos de Trabajo
		● Contratos de Venta
		● Documentación de Contabilidad
		● Documentación de Propuesta
		● Documentación Legal
		● Factura de Compra
		● Factura de Venta
		● Manuales
		● Oficios
		● Otros Documentos

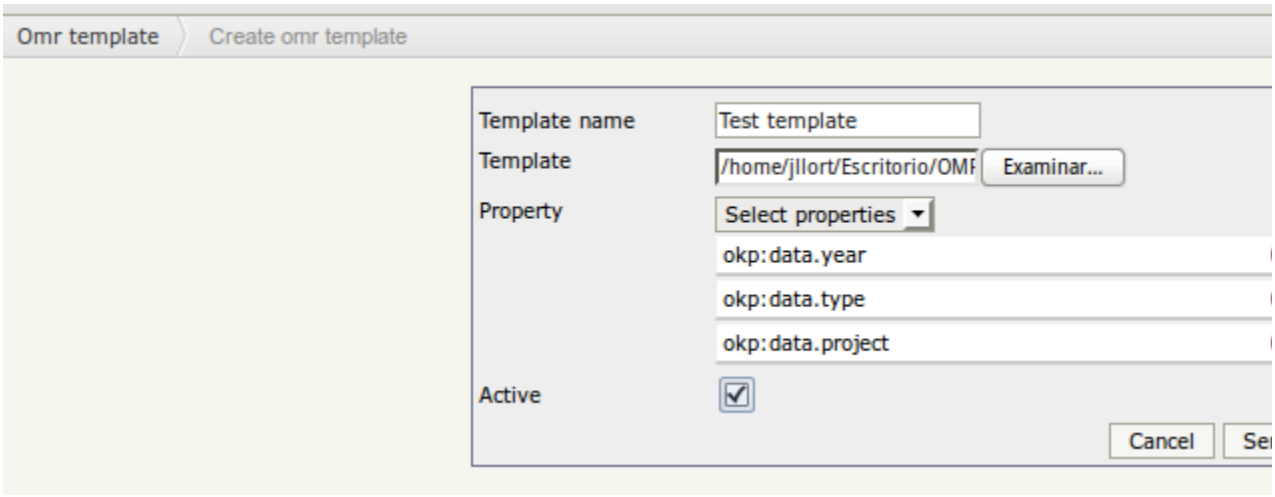
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create a new template

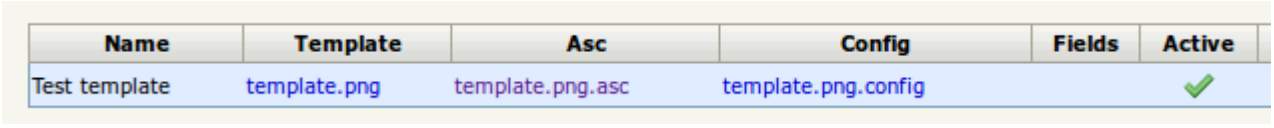
- *On top right click on* ***Create new template icon***.
- ***Choose the template image*** *produced in preliminar step.*
- *Choose the* ***metadata fields***.

> *More information about register Metadata.*

- *Click on* ***Send button***.



> *The application will generate two training files (.asc and .config) based on template file name:*
>
> - *template_file_name.asc*
> - *template_file_name.config*

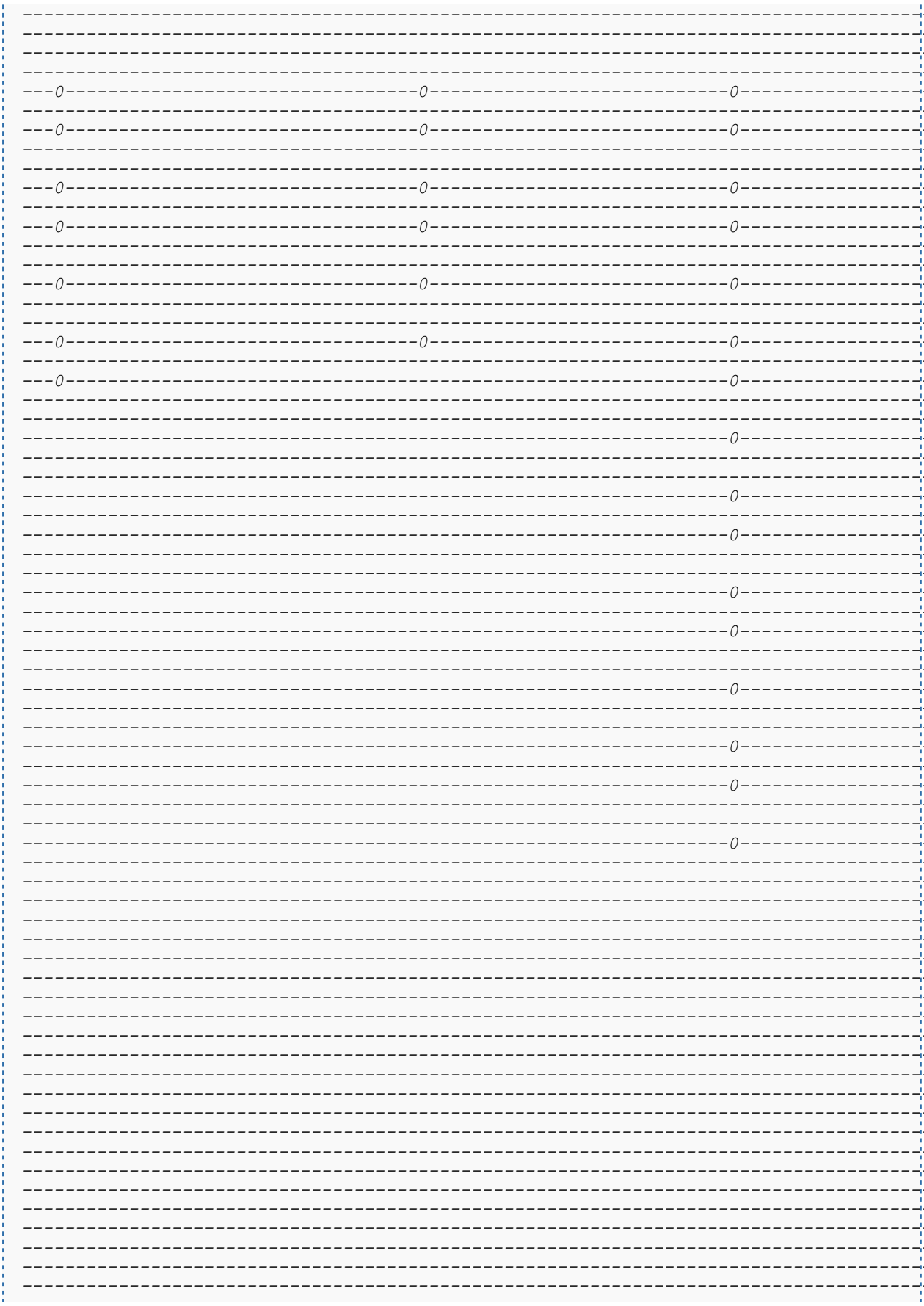


Name	Template	Asc	Config	Fields	Active
Test template	template.png	template.png.asc	template.png.config		✓

Update asc training files

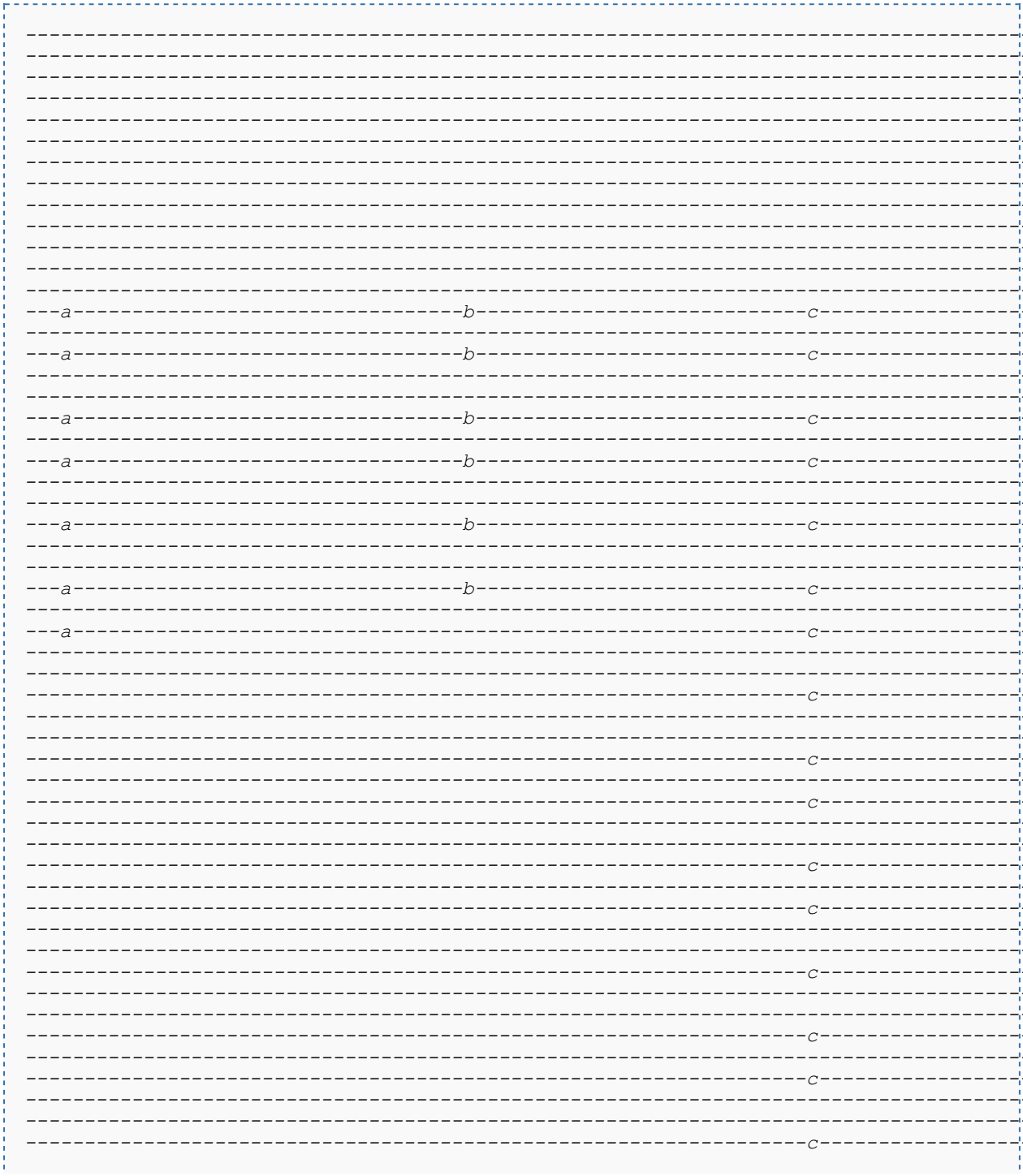
Download the tranining .asc file and openkm with a standard text editor. This is actually an ascii representation of the template with all the marks marked with a '0', as shown below.

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```

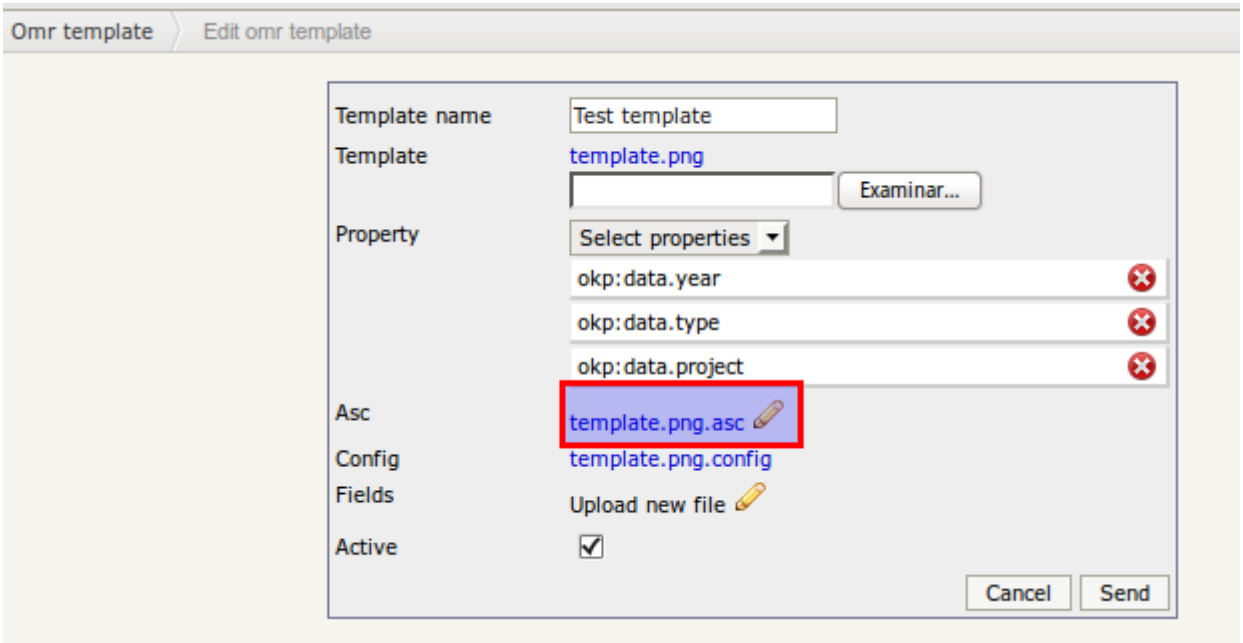


Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Some of the marks may not be exactly aligned, but don't worry. Now, very carefully, label the marks with single characters, by setting all the marks in the same group with the same character. This is shown below. In case you run out of alphabets, you can use numerals or any other character such as - + ? . etc. Make sure that you do not alter the position of any mark! This is agreably hard to do, but we chose this method to avoid the need for making a graphical frontend.



- *Click on* ***Edit template icon***.
- *Click on* ***Edit ".asc" training file icon***.
- ***Chose the training .asc file***.
- *Click on* ***Send button***.



Update .fields training files

You now need to make a [template-filename].fields file, which contains the details of the various form-fields, as follows.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The format of each line is: character type subtype name value1 value2....

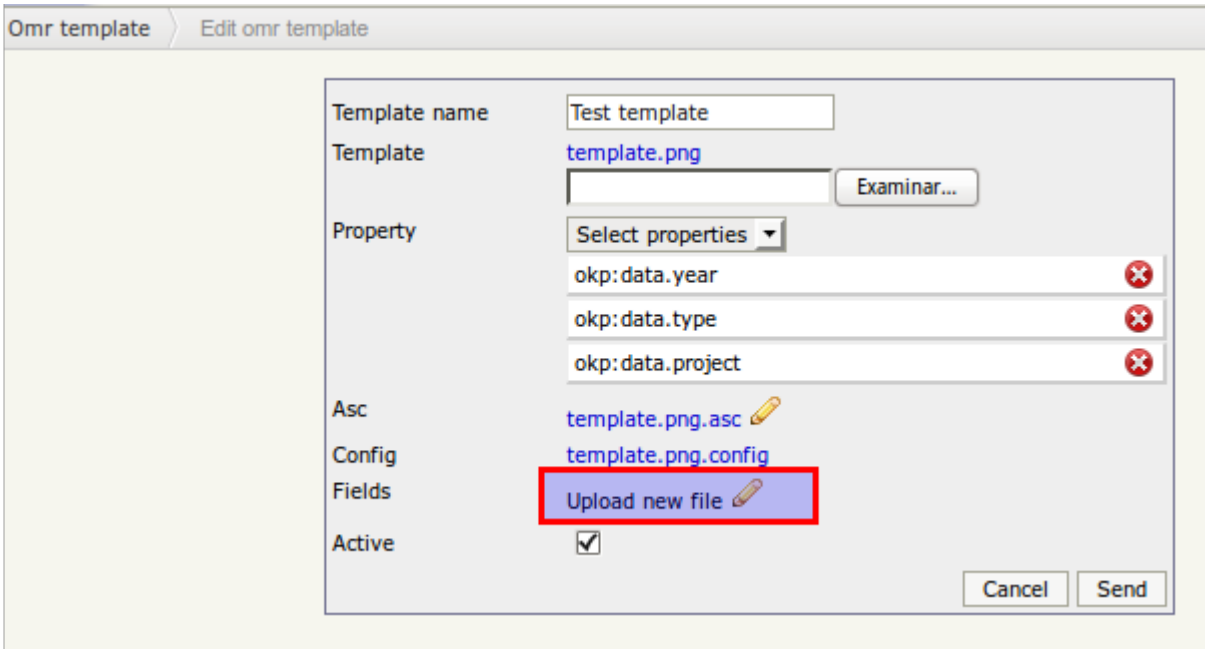
Field	*Description*
Character	*This is the same as the character you used in the .asc file*
Type and Subtype	*The types of fields supported are single and multiple choice lists, aligned row-wise (horizontally) or column-wise (vertically). And grids aligned horizontally or vertically. Each field is denoted by (i) row single or row multiple for a single or multiple selection field aligned row-wise, (ii) column single or column multiple for a single or mutiple selection field aligned column-wise, or (iii) grid row or grid column for a grid field aligned row-wise or column-wise respectively.*
Field name	*This can be any name you want to associate with the field.*
Values	*These are the field values, in order.* *The values has a direct corresponde with metadata field values, first indicate the field name and others the values into*

Example:

```
a column single okp:data.project alarmasCalleLarga alarmasRinconada alcantarilladosRi
b column single okp:data.year 2012 2013 2014 2015 2016 2017
c column single okp:data.type actas boletaCompra boletaHonorario boletaVenta comproba
```

- *Click on* ***Edit template icon***.
- *Click on* ***Edit fields training file icon***.
- ***Chose the training .fields file***.
- *Click on* ***Send button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

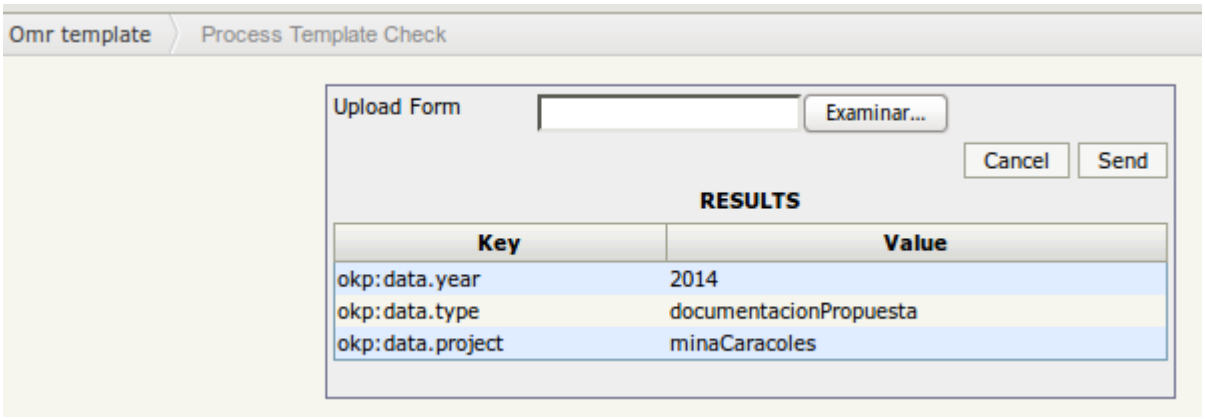


Check the template

- *Click on* ***Check icon****.*
- ***Choose some file to upload*** *(will be used to capture data).*

Seleccione Proyecto	Seleccione Año del Documento	Seleccione Tipo de Documento
● Alarmas Calle Larga	● 2012	● Actas
● Alarmas Rinconada	● 2013	● Boleta de Compra
● Alcantarillados Rinconada	● 2014	● Boleta de Honorario
● Mina Caracoles	● 2015	● Boleta de Venta
● Colegio Metodista	● 2016	● Comprobante de Compra
● Obras Adicionales MC	● 2017	● Comprobante de Venta
● Villa Las Rosas		● Contratos de Trabajo
		● Contratos de Venta
		● Documentación de Contabilidad
		● Documentación de Propuesta
		● Documentación Legal
		● Factura de Compra
		● Factura de Venta
		● Manuales
		● Oficios
		● Otros Documentos

And you'll get the results:



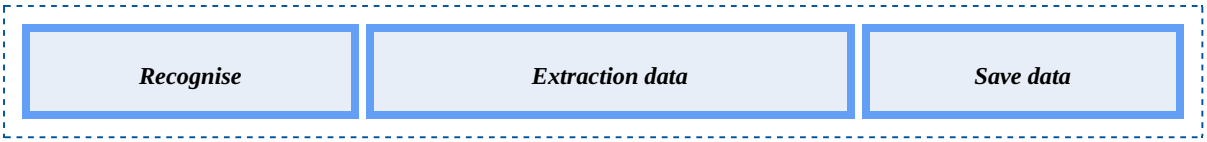
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OCR templates

OCR Templates allow to create zonal OCR templates which allows to recognise and extract estructured text from scanned images.

For best results, images should be scanned at least at 200 dpi resolution to get good text recognition from the OCR engine.

Understanding stages on the OCR templates execution:



On recognition, the application based on ocr template's definition tries to identify the document. A document will be considered recognised only if an OCR template checks 100%

On extraction data, the values are extracted from the document.

On save data, the dynamic values got in the extraction data stage are saved into metadata groups. Optionaly are saved as static values defined on the OCR template (normally used to identify the document type).

There're two field types on OCR templates:

- *Validation Control fields.*
- *Data Capture fields.*

***Validation Controls fields** are used only on the **recognition** stage.*

***Data Capture fields** are used on the **extraction data** stage and optionaly can be used also on **recognition** stage.*

Allowed MIME types on the recognise stage are:

- *PNG*
- *JPG*
- *GIF*
- *BMP*
- *PDF*

Validation Control field description

Validation Control fields are used only in the Recognise stage in helping on the document identification by checking a document's section pattern.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Each Validation Control field is composed of:

Field	*Description*	*Required*
Name	*Name of the field.*	*true.*
Order	*Order of the evaluation*	*true.*
Type	*List of **OCR Validation Controls**. Each validator is a condition that must be complied.*	*true.*
Value	*A value.*	*
Pattern	*A Java regex pattern.*	*
Rotation	*Indicates the image rotation that must be applied.* *Available image rotations:* • *-270* • *-180* • *-90* • *0* • *90* • *180* • *270*	*true.*
Custom	*Sets commands to process images by the OCR. Several commands can be set , one per line.* *For example to highlight the red color* ***/usr/bin/convert ${fileIn} -matte (+clone -fuzz 50% -transparent red) -compose DstOut -composite -negate ${fileOut}***	*false.*
	OCR command line. *When present it will be executed in place of the system.ocr configuration parameter.*	

OCR	*More information Configuration parameters.* *For example to force digits capture.* ***/usr/bin/tesseract ${fileIn} ${fileOut} digits***	*false.*
Empty value allowed	*When checked indicates ocr empty text value is also valid in identifying document process.*	*false*
***** These parameters are used into to the Validation Control. Can be required or not depending on which Validation Control is used.*		

Data Capture field description

Data Capture fields are used to retrieve data in the extraction stage but also can be used in the Recognise stage in helping a document identification by checking a document section pattern.

Each Data Capture field is composed of:

Field	*Description*	*Required*
Name	*Name of the field.*	*true.*
Order	*Order of evaluation*	*true.*
Type	*List of the* ***OCR Data Capture***.	*true.*
Property	*Metadata field where will be stored the captured data.*	*true.*
Pattern	*A Java regex pattern.*	*
Rotation	*Indicates the image rotation that must be applied.* *Available image rotations:* • *-270* • *-180* • *-90*	*true.*

	• *0* • *90* • *180* • *270*	
Custom	*Sets commands to process the images by the OCR. Several commands can be set, one per line.* *For example to highlight the red color* ***/usr/bin/convert ${fileIn} -matte (+clone -fuzz 50% -transparent red) -compose DstOut -composite -negate ${fileOut}***	*false.*
OCR	*OCR command line.* *When present it will be executed in place of the system.ocr configuration parameter.* *More information Configuration parameters.* *For example to force digits capture.* ***/usr/bin/tesseract ${fileIn} ${fileOut} digits***	*false.*
Use to recognize	*When it's set to true, indicates that the field will also be used on the recognise stage.*	*false.*
***** This parameter is used in the Data Capture. It can be required or not depending on which Data Capture is used.*		

Understanding OCR Validation Control and OCR Data Capture

The application provides a complete list of available OCR Validation Controls and OCR Data Capture from Plugins view.

Application plugins architecture help the administrators to add new features to the existing application.

By default application comes with built in plugins:

- *Built in OCR Validation Control plugins.*
- *Built in OCR Data Capture plugins.*

Also your own plugins can be created:

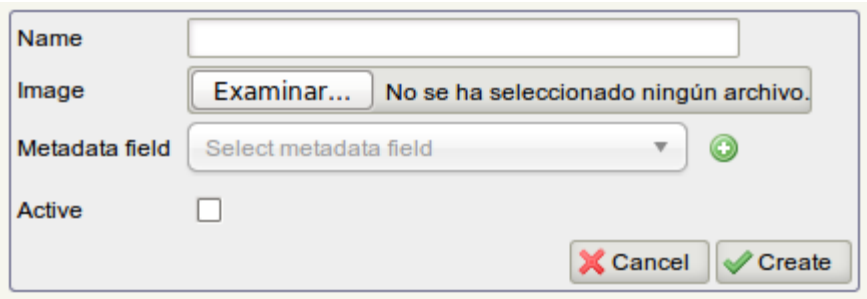
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

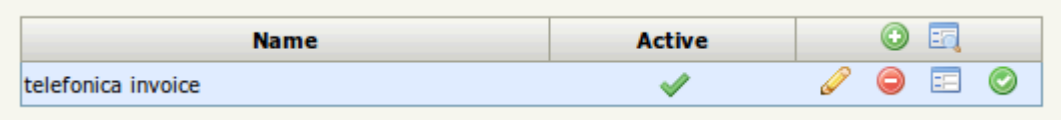
- *Creating your own OCR Validation Control plugin*.
- *Creating your own OCR Data Capture plugin*.

> *For more information about how to Register a new plugin.*

Create a new OCR template

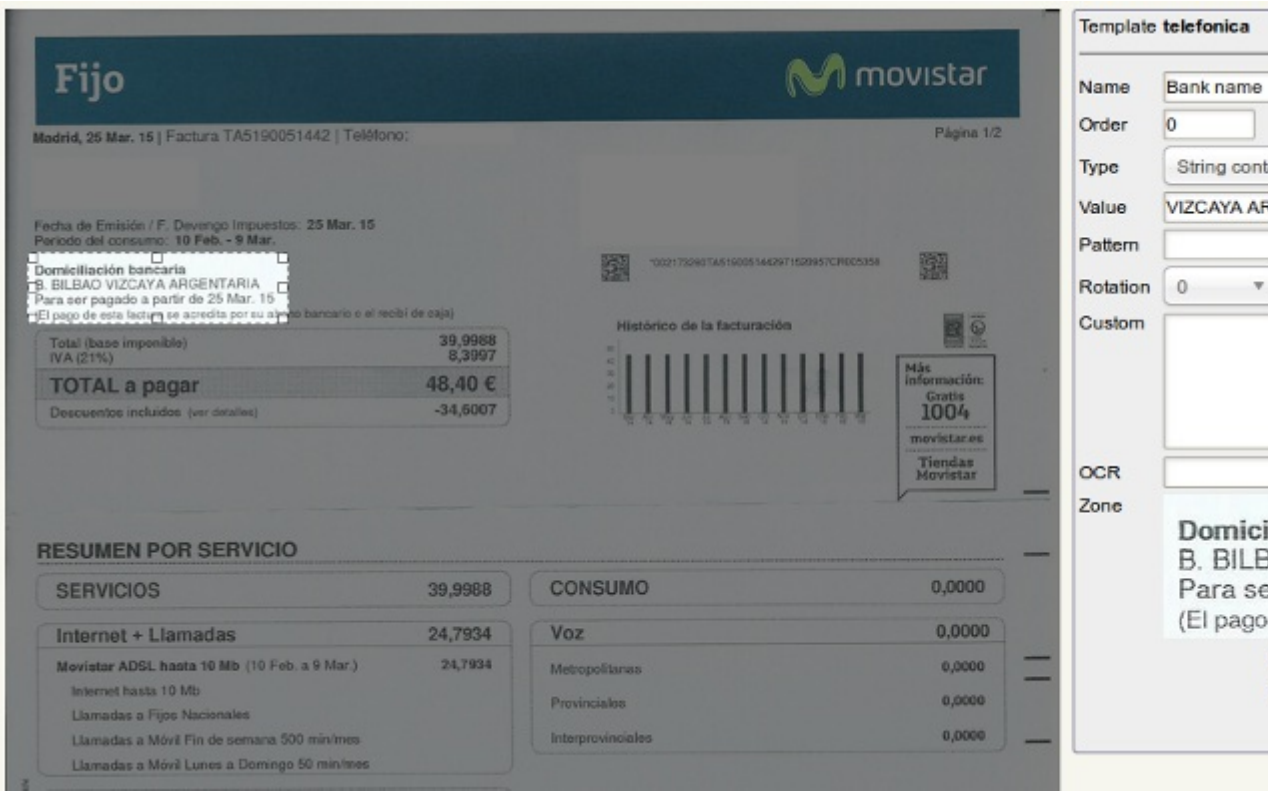
- *On the top right click* ***New OCR templace icon***.
- ***Fill in the fields***.
  - *Metadata field (static metadata values, normally used to identify document type).*
- *Chose the file (allowed MIME type).*
- *Click on* ***Create button***.





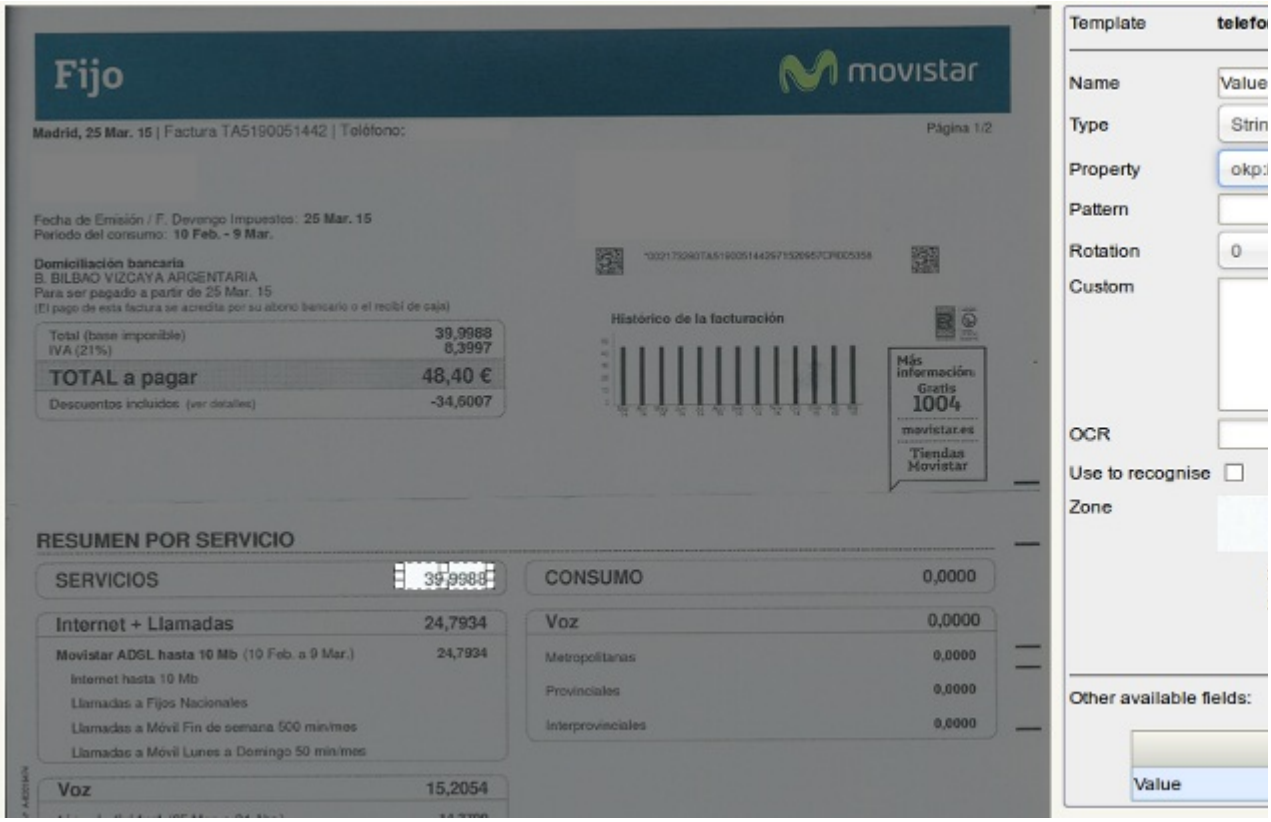
- *Add Validation Controls fields according to your convenience.*
  - *Validation Controls*
    - *Click on* ***Fields***.
      - *To create a new Validation Control:*
        - *Click on* ***New Validation Control Field icon***.
        - ***Fill the fields***.
        - ***Choose the area with mouse on the left image***.
        - *Click on* ***Create button***.
      - *To edit a Validation Control:*
        - *Click on* ***Edit a Validation Control icon***.
        - ***Modify the fields***.
        - *Click on* ***Edit button***.
      - *To delete a Validation Control:*

- *Click on* ***Delete a Validation Control icon***.
- *Click on* ***Delete button***.



- *Add Data Capture fields according to your convenience.*
  - *Data Capture*
    - *Click on* ***Fields***.
    - *To create a new Data Capture*
      - *Click on* ***New Data Capture Field icon***.
      - ***Fill in the fields***.
      - ***Choose the area with mouse on the left image***.
      - *Click on* ***Create Button***.
    - *To edit a Data Capture:*
      - *Click on* ***Edit a Data Capture icon***.
      - ***Modify the fields***.
      - *Click on* ***Edit button***.
    - *To delete a Data Capture:*
      - *Click on* ***Delete a Data Capture icon***.
      - *Click on* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Edit a OCR template

- *Click on* ***Edit OCR template icon***.
- ***Modify de fields***.
- *Click on* ***Edit button***.

Edit a OCR template fields

Edit a OCR Validation Control

Click on ***Fields***.

- *Click on* ***Edit a Validation Control icon***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Edit a OCR Data Capture

Click on ***Fields***.

- *Click on* ***Edit a Data Capture icon***.
- ***Modify the fields***.
- *Click on* ***Edit button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Delete a OCR template fields

Delete a OCR Validation Control

Click on ***Fields***.

- *Click on* ***Delete a Validation Control icon***.
- *Click on* ***Delete button***.

Delete a OCR Data Capture

Click on ***Fields***.

- *Click on* ***Delete a Data Capture icon***.
- *Click on* ***Delete button***.

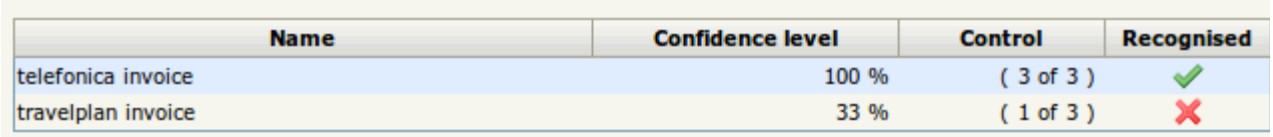
Check OCR template

- *Click on Check icon and the OCR template will be checked.*

> *In the Capture Data field is shown the result of the text extraction by the OCR.*
>
> *In the Validation Control field is shown the result of the compliance validation (true or false).*

Check recognise a document

- *On top right click on* ***Recognise icon***.
- ***Choose the file to be recognised*** *(allowed MIME type).*
- ***Choose the recognise level***.
- *Click on* ***Recognise button***.



Name	Confidence level	Control	Recognised
telefonica invoice	100 %	(3 of 3)	✓
travelplan invoice	33 %	(1 of 3)	✗

> *Recognise level, use more or less speed to identify a document. By default application uses slow.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *In the fast level, it takes only in consideration the Validation Control fields.*
- *In the medium level, it also takes in consideration that the Data Capture fields must not be empty.*
- *In the slow level, it also takes in consideration that the Data Capture fields are right (complish with patterns etc...).*

Delete a OCR template

- *Click on* ***Delete OCR template icon***.
- *Click on* ***Delete button***.

Enable OCR Data Capture

Feature	*Description*
Profile General tab	*When the OCR data capture is enabled from the Profile General tab, the OCR Data Capture process will be executed automatically on each new uploaded document.* *In case that is present an automation definition, these parameters have no effect.* *With the automation feature all these cases and more are covered.*
Automation	*Validations:* • ***Is OCRDataCaptureFile*** *More information about Built-in Automation Validator plugins.* *Actions:* • ***OCR DataCapture*** • ***Add OCRDataCaptureToWizard*** *More information about Built-in Automation Action plugins.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Built in OCR Validation Control plugins

Name	*Value*	*Pattern*	*Description*
String contains	*Optional*	*Optional.*	*Returns the evaluation of a "contains operation" with a value or a pattern within the text extracted by OCR.* *In case the pattern field is empty and the value field is not empty, it returns true if the text extracted by OCR contains the value of the value field.* *In case the pattern field is not empty and the value field is empty, it returns true if the text extracted by OCR matches the pattern only one time.* *In case the pattern field is not empty and the value the field is not empty, it returns true if the text extracted by OCR matches the pattern only one time and the matched pattern contains the value field.* *In case the pattern and value fields are empty, it returns false.*
String equals	*Optional.*	*Optional.*	*It returns the evaluation of equal operations with a value or a pattern with the text extracted by OCR.* *In case the pattern field is empty and the value field is not empty, it returns true if the text extracted by OCR equals the value of the value field.* *In case the pattern field is not empty and the value field is empty, it returns true if the text extracted by OCR matches the pattern only one time.* *In case the pattern field is not empty and the value field is not empty, it returns true if the text extracted by OCR matches the pattern only one time and the matched pattern is equal to the value field.* *In case the pattern field is empty and the value field is empty, it returns false.*
Similar	*Optional.*	*Optional*	*Returns the evaluation of similar operations with some values or a pattern with the*

String			*text extracted by OCR.* *The "similar operation" counts the number of changes that must be done to transform the expected value with the text extracted by ocr. This is the **distance between both texts**.* *This plugin works in combination with a configuration parameter named ocr.control.similar.distance. By default the value of this parameter is 5.* *In case a pattern field is empty and the value field is not empty, it returns true if the distance between the text extracted by OCR and the value of the value field is less than the one set in ocr.control.similar.distance parameter.* *In case a pattern field is not empty and the value field is empty, it returns true if the text extracted by OCR matches the pattern only one time.* *In case a pattern field is not empty and the value field is not empty, it returns true if the text extracted by OCR matches the pattern only one time and the distance between the matched text and the value of the value field is less than set in ocr.control.similar.distance parameter.* *In case a pattern field is empty and value field is empty, it returns false.*

Built in OCR Data Capture plugins

Name	*Pattern*	*Description*
String	*Optional.*	*Return a text value.* *In case the pattern is empty returns text extracted by OCR.* *In case the pattern is not empty, returns the text extracted by OCR that matches the pattern only one time.* *In case the text extracted by OCR is empty, it returns and Exception.* *In case the pattern is not empty and the text extracted by OCR doesn't match the pattern only one time,it returns an Exception.*
Number	*Optional.*	*Return a number.* *In case the pattern is empty, it returns a number based on the text extracted by OCR.* *In case the pattern is not empty,it returns a number based on the text extracted by OCR that matches the pattern.* *In case the text extracted by OCR is empty, it returns an exception.* *In case the pattern is not empty but the text extracted by OCR doesn't match the pattern, it returns an exception.* *In case the text extracted can not be converted into a number, it returns an exception.*
Date	*Mandatory.*	*Returns a Calendar value.* *In case the pattern is not empty, it returns a Calendar based on the text extracted by OCR and the pattern.* *In case the pattern is empty returns an exception.* *In case the text extracted by OCR is empty it returns an exception.* *In case the text extracted can not be converted to a Calendar, it returns an exception.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

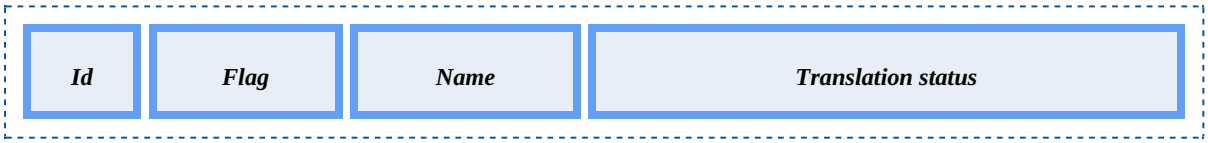
Barcode	*Optional.*	*Return barcode text.* *In case the pattern is empty, it returns a text extracted by a Barcode engine.* *In case the pattern is not empty, it returns the text extracted by a Barcode engine that matches the pattern.* *In case the text extracted by a Barcode engine is empty, it returns an exception.* *In case the pattern is not empty but text extracted by a Barcode engine doesn't match the pattern, it returns an Exception.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Languages

The application can work with several languages, here are set the languages terms definitions.

Languages are shown in a table structure.



The "en-GB" translation is used by application as default translation.

Sometimes translations are not fully completed and some terms are missing. The "en-GB" is used to identify which are the missing terms and how many are.

When a term is missing in other language the "en-GB" term is used in its place.

en-GB translation can not be removed.

Create a new language

- *Click on* ***new language icon***.
- *Fill the fields*
  - *The id must be compliant with ISO 639 (2 levels).*
  - *Language name*
  - *The icon should be 16x11 px*
- *Click on* ***Create button***.

More information about ISO 639.

Edit a language

- *Click on* ***Edit language icon***.
- ***Change the fields***.
- *Click on* ***Edit button***.

Delete a language

- *Click on* ***Delete language icon***.
- ***Change the fields***.
- *Click on* ***Delete button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Edit language translations

On the right top the translation can be filtered by key.

- *Click on* ***Edit language translation icon***.

Languages translations are shown in a table structure.

Module	Key property	English translation	Translation

By default are ***tree modules*** *that indicate where translations are used:*

- *frontend (used on frontend)*
- *mobile (used on mobile view)*
- *extension (used on frontend by extension modules)*

Two key property can not be equals.

- ***Change the fields***.
- *Click on* ***Edit button***.

Export a language translations

- *Click on* ***Export language translations icon***.
- *Application* ***retrieve a SQL file*** *with language translation into.*

Import a language translations

Download your language package from Language translations.

- *On right bottom* ***choose a Language package file***.
- *Click on* ***Add new translation button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Language translations

Language	*File*	*Number of terms*
Arabic	***OpenKM_7.1_ar-SD.sql***	*670*
Arabic KSA	***OpenKM_7.1_ar-SA.sql***	*1892*
Azeri	***OpenKM_7.1_az-AZ.sql***	*1565*
Belgian/Flemish	***OpenKM_7.1_nl-BE.sql***	*911*
Bosnian	***OpenKM_7.1_bs-BS.sql***	*584*
Bulgarian	***OpenKM_7.1_bg-BG.sql***	*909*
Català	***OpenKM_7.1_ca-ES.sql***	*1199*
Chinese simple	***OpenKM_7.1_zh-CN.sql***	*1275*
Chinese traditional	***OpenKM_7.1_zh-TW.sql***	*1279*
Czech	***OpenKM_7.1_cs-CZ.sql***	*1199*
Danish	***OpenKM_7.1_dk-DK.sql***	*884*
Deutsch	***OpenKM_7.1_de-DE.sql***	*1649*
Deutsch - Austria	***OpenKM_7.1_de-AT.sql***	*903*
Dutch	***OpenKM_7.1_nl-NL.sql***	*1334*
English	***OpenKM_7.1_en-GB.sql***	*1605*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

English - USA	***OpenKM_7.1_en-US.sql***	*585*
Español	***OpenKM_7.1_es-ES.sql***	*1605*
Español Colombia	***OpenKM_7.1_co-ES.sql***	*584*
Euskera	***OpenKM_7.1_eu-ES.sql***	*1565*
Farsi	***OpenKM_7.1_fa-FA.sql***	*584*
French	***OpenKM_7.1_fr-FR.sql***	*1572*
Gallego	***OpenKM_7.1_gl-ES.sql***	*584*
Greece	***OpenKM_7.1_el-GR.sql***	*1205*
Hungarian	***OpenKM_7.1_hu-HU.sql***	*585*
Indonesian	***OpenKM_7.1_id-ID.sql***	*585*
Italian	***OpenKM_7.1_it-IT.sql***	*1415*
Japanese	***OpenKM_7.1_jp-JP.sql***	*890*
Korean	***OpenKM_7.1_ko-KR.sql***	*1902*
Latvian	***OpenKM_7.1_lv-LV.sql***	*585*
Lithuanian	***OpenKM_7.1_lt-LT.sql***	*584*
Macedonian	***OpenKM_7.1_mk-MK.sql***	*585*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Norwegian	***OpenKM_7.1_nb-NO.sql***	*1175*
Polish	***OpenKM_7.1_pl-PL.sql***	*1286*
Português do Brasil	***OpenKM_7.1_pt-BR.sql***	*1436*
Romanian	***OpenKM_7.1_ro-RO.sql***	*585*
Russian	***OpenKM_7.1_ru-RU.sql***	*1242*
Serbian	***OpenKM_7.1_sr-SR.sql***	*942*
Slovak	***OpenKM_7.1_sk-SK.sql***	*1447*
Suomi	***OpenKM_7.1_fi-FI.sql***	*1103*
Swedish	***OpenKM_7.1_sv-SE.sql***	*585*
Thai	***OpenKM_7.1_th-TH.sql***	*585*
Turkish	***OpenKM_7.1_tr-TR.sql***	*1259*
Vietnamese	***OpenKM_7.1_zh-VN.sql***	*1227*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

How to create a new translation

Before starting with a new translation, please check if is available in Language translations section.

A sample of new language base on Oromo from Ethiopia

*The first step is identifying what is your language code based on ISO 639 (2 level). Looking at the table of language identifiers, the code might be "**om-ET**".*

*The second step will get the country flag, in this case, Ethiopia. Go to famfamfam.com website and download the flags icons. The flags icon name are based ISO 369 (one level), that means we are looking for a file named "**et.gif**".*

The third step will be to create the new language. Go to Administration > Languages and create a new language with these values:

- *Click on* ***new language icon***.
- *Fill the fields*
  - *The "id" might be "**om-ET**".*
  - *The "name" might be "Oromo".*
  - *The "icon" might be the file named "**et.gif**".*
- *Click on **Create button**.*

Once the new language has been created must set the translations:

- *Click on* ***Edit language icon***.
- ***Change the fields***.
- *Clic,k on **Edit button**.*

Finally, think on sharing with the community your translation.

- *Click on* ***Export language translations icon***.
- *The application **retrieves a SQL file** with language translation into.*
- *Contact us at the contact website form and we will tell you the way for sharing with us the translation.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Utilities

The utilities section at Administration is a collection of useful features designed to be used by IT staff.

To take advantage of most of these feature some technical skills are needed .

These features can be categorized as:

Category	*Description*
Security	• *Admin permissions* • *Download tokens* • *Purge permissions* • *Purge trashes*
Performace	• *Cache stats* • *Clean conversion cache* • *Hibernate stats* • *Profiling stats*
Low level access	• *Database query* • *Execute JAR* • *Scripting*
Check specific configuration	• *Check email* • *Check text extraction* • *LDAP Search* • *System properties* • *Create support zip*
Import & export	• *Repository import* • *Repository export*
Search engine	• *List indexes*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	• *Rebuild indexes*
Others	• *Folder style* • *Plugins* • *Repository checker* • *Thesaurus*
Experimental features	• *LogCat*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Admin permissions

The Admin permission feature helps the administrators to set the Administration access levels to users who have access to the application.

For example, it provides access to some users only for managing users and roles.

On this view it is shown a table with all the users who have access to administration based on admin role.

More information about default admin role at Security configuration parameters.

Change user level access

- *Click on* ***Edit user level access icon***.
- ***Change level access***.
- *Click on* ***Edit button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

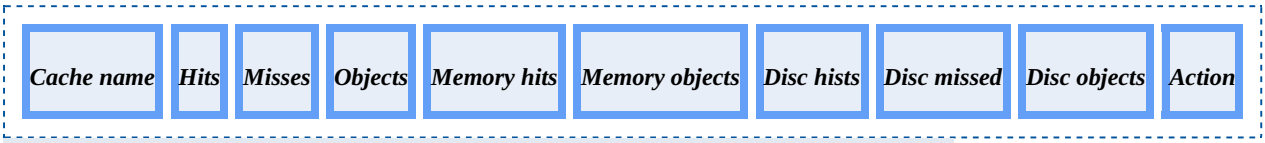
Cache stats

OpenKM takes advantage of cached objects for fast retrieving data.

> *For example the users and roles list, when application is configured to working with LDAP, are cached to reduce the traffic between application and LDAP server.*
>
> *The associated cached name for LDAP are:*
>
> - *com.openkm.cache.ldapPrincipalAdapter.general*
> - *com.openkm.cache.ldapPrincipalAdapter.name*

Also cache can be used to collect data useful to find out configuration parameters changes needed for better application perfomance. For this purpose the application comes with a cache audit trail.

Cache stats are shown in a table structure:



Field	Description	Audit trail required
Cache name	*Unique class name of the cached class.*	*Not applicable.*
Hits	*Number of successful queries.*	*true*
Misses	*Number of failed queries.*	*true*
Objects	*Number of cached objects.*	*false*
Memory hits	*Number of successful queries in cache memory.*	*true*
Memory missed	*Number of failed queries in cache memory.*	*true*
Memory objects	*Number of objects in cache memory.*	*false*
Disc hits	*Number of successful queries in cache disk.*	*true*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Disc missed	*Number of failed queries in cache disk.*	*true*
Disc objects	*Number of objects in cache disk.*	*false*
Action	*Clean cached objects.*	*Not applicable.*

Clean all cache

- *On the right top click on* ***Reset all button***.

Refresh cache stats

- *On the top right click on* ***Refresh icon***.

Enable cache stats audit trail

- *On the top right click on* ***Enable cache audit trial button***.

Enables the fields stats:

- *Hits.*
- *Misses.*
- *Memory Hits.*
- *Memory missed.*
- *Memory objects.*
- *Disc hits.*
- *Disc missed.*

Disable cache stats audit trail

- *On top right click on* ***Disable cache audit trial button***.

Disable the fields stats:

- *Hits.*
- *Misses.*
- *Memory Hits.*
- *Memory missed.*
- *Memory objects.*
- *Disc hits.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Disc missed.*

Clean cache stats audit trail

- *On top right click on* ***Clear cache audit trial button***.

Reset to zero the value of the fields stats:

- *Hits.*
- *Misses.*
- *Memory Hits.*
- *Memory missed.*
- *Memory objects.*
- *Disc hits.*
- *Disc missed.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Check email

The check email feature helps the administrator to check the application mail configuration.

Check email

- *Fill the fields*
  - *From*
  - *To*
  - *Subject*
  - *Content*
- *Click on **Check button***

If the application mail configuration application is correctly set it will return an "OK" otherwise will be shown the cause of the error.

*If the **send.mail.from.user** configuration property is set to **false**, the "From" field will be ignored and will make use of the **mail.from** JavaMail configuration property. Note that depending on the SMTP server configuration only the address configured in **mail.smtp.user** may be able to send messages, this mean, the user who appears in the "From" field.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Check text extraction

This feature helps the administrator to check the text extraction process.

It can check the extraction process based on:

- *Document to be processed*
  - *Document UUID (see tab properties when document is selected on general application user interface).*
  - *Document path.*
  - *Document outside application (choose a document from file system to be uploaded).*
- *List of available text extractors.*

Check text extraction

- ***Choose a document** to be processed*
  - *OpenKM document:*
    - *Fill field document UUID or document path.*
  - *External document:*
    - *Choose a document from file system to be uploaded.*
- ***Choose one text extractor** from available text extractors list.*
- *Click on **Check button**.*

After the process is completed it will show the text extracted by the selected text extractor.

The text extraction process can take some seconds or minutes depending on the size of the document to be processed and the file MIME type.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Clean conversion cache

The application comes with integrated file conversion cache.

File conversion cache is stored in the file system, by default into $TOMCAT/repository/cache folder.

A classical example of document conversion is generating the thumbnail of each document shown in the default application user interface. For example to get a thumbnail of a Microsoft Word document is needed convert the first page as an image. For doing this process application consumes some CPU.

To prevent it consumes more CPU than necessary, the documents are converted only when it's necessary, first time, or after document update. In other cases the file is retrieved from cached files.

The administrator can take control of these cached files and clean by type or all.

Cached files:

Type	*Description*
DXF	*Caused by autocad pre visualization conversion.*
PDF	*Several causes:* • *Document pre visualization.* • *User conversion by demand.*
SWF	*Document pre visualization.*
Thumbnail properties	*Document thumbnail in application desktop view.*
Thumbnail Search	*Document thumbnail in application search view.*
Thumbnail Lightbox	*Document thumbnail in lightbox.*

Clean conversion cache

- ***Choose cached file to be cleaned***
- *Click on* ***Clean button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The operation can take few seconds or minutes depending the number of files to be removed from the file system.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Database query

This feature helps administrator accessing database data, easily and quickly without thirdparty database software client.

The results of the queries can be exported to SQL or CSV file formats.

Allowed types:

Type	*Description*	*List*	*Export*
JDBC	*SQL queries supported by the configured database.*	*When choosen is shown a list of available tables.*	*Yes.*
Hibernate	*Neutral database queries based on hibernate queries format.*		*No.*

- *Is allowed writing more than one query.*
- *Each query must be written on a single line, break lines are not allowed.*
- *Execute will process all written queries in same order has been written, top to bottom. Queries are executed as a FIFO (first in first out).*
- *In case some query fails, that will not stop the execution of the others.*

Execute JDBC query

- *On bottom right **choose type JDBC**.*
- *Write some valid SQL.*
- *Click on **Execute button**.*
- *Optionaly the results can be exported to SQL or CSV*
  - *Click on Export to sql icon.*
  - *Click on Export to csv icon.*

Example:

```
// nodeBase = documents + folders + mails + records
SELECT count(*) from OKM_NODE_BASE;
SELECT count(*) from OKM_NODE_DOCUMENT;
SELECT count(*) from OKM_NODE_FOLDER;
SELECT count(*) from OKM_NODE_MAIL;
SELECT count(*) from OKM_NODE_RECORD;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
// select first 10 records
SELECT * from OKM_NODE_BASE LIMIT 10;
SELECT * from OKM_NODE_DOCUMENT LIMIT 10;
SELECT * from OKM_NODE_DOCUMENT_VERSION LIMIT 10;
SELECT * from OKM_NODE_FOLDER LIMIT 10;
SELECT * from OKM_NODE_MAIL LIMIT 10;
SELECT * from OKM_NODE_RECORD LIMIT 10;
```

Execute Hibernate query

- *On bottom right **choose type Hibernate**.*
- *Write some valid HQL.*
- *Optionally you can check the "Show SQL" option.*
- *Click on **Execute button**.*

When "Show SQL" option is checked is shown in results the real sql executed.

Example:

```
select count(*) from NodeBase;
select count(*) from NodeFolder;
```

Allowed sentences

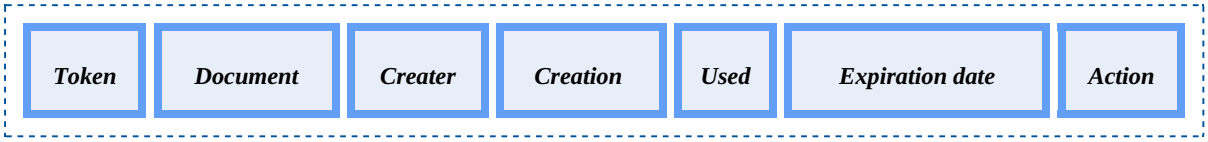
- *On bottom right **choose a valid sql file**.*
- *Click on **Import SQL script button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Download tokens

The download token feature helps application users to share documents with external users without need to be logged into the application. With Download tokens log feature, the administrator can take control over the documents that has been shared, the users who shared it and when had been downloaded.

Download tokens log are shown is a table structure:



ï¿½

Field	Description
Token	*The unique token UUID.*
Document	*The shared document.*
Creater	*Id of the user who shared the document.*
Creation	*Date when the document had been shared.*
Used	*Indicates if the document has been downloaded or not.*
Expiration time	*Indicates the date when the document will not be longer downloaded.*
Action	*Deletes a token*

Delete token

- *Click onï¿½* **Delete token button**.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Execute JAR

A JAR file is a type of compressed file, which is actually using the same compression algorithms and format as ZIP files.

JAR files normally contain a number of Java Classes and some metadata (a manifest file). Java classes are part of applications which are written in Java.

This feature helps administration to create its own classes, packaged into JAR file and execute from OpenKM.

Examples of usage:

- *When administrator wants to do some operation across all repository that take effect on a lot of nodes, based on some logic or special conditions. Can take advantage building minimal code with OpenKM API for it. Some examples could be, change security, remove or move files, rename files based on their metadata values, among others.*
- *Check crontab task.*

Requisits:

- *Must implement the class "**public static void main(String[] args)**".*

```
package com.openkm;

public class CronTabTest {
    public static void main(String[] args) {
        // Some stuff here
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Folder style

The application changes the folder's icons based on the security granted, users subscription folder has subfolders into. There's about 8 possible folder status.

Available folder status:

Has write Grants	*Has subfolders*	*Is user Subscribed*	*Name*	*Icon*
No.	*No.*	*No.*	*Read-only icon.*	
	Yes.	*No.*	*Read-only with children icon.*	
	No.	*Yes.*	*Read-only with subscription icon.*	
	Yes.	*Yes.*	*Read-only with children and subscription icon.*	
Yes.	*No.*	*No.*	*Read-write icon.*	
	Yes.	*No.*	*Read-write with children icon.*	
	No.	*Yes.*	*Read-write with subscription icon.*	
	Yes.	*Yes.*	*Read-write with children and subscription icon.*	

The folder style named "default" can not be deleted or renamed.

The folder style named "default" is used by application to set thefolder icons on all the folders by default.

Advanced sample information about this subject at: Change folder style by metadata value.

Create a new Folder style

- *On top right click on* ***New folder style icon***.
- ***Set the name***.
- *Click on* ***Create button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Repeat Click on* ***Create icon*** *for all empty icon status, until all statuses are allocated an icon.*
  - ***Choose an image*** *from your file sytem.*
  - *Optionally check "Copy to all icons" if you want to use the icon to all icon status.*
  - *Click on* ***Create button****.*
- *Finally click on* ***Done button****.*

The size of the images should be ***16px x 16px****.*

Two style folders can not have the same name.

Clone a Folder style.

- *Click on* ***Clone icon****.*
- ***Set a new name****.*
- *Click on* ***Clone button****.*

Edit a Folder style

- *Click on* ***Edit a folder style icon****.*
  - *Click on* ***Edit icon****.*
  - ***Choose an image*** *from your file sytem.*
  - *Optionally check "Copy to all icons" if you want to use the icon to all icon status.*
  - *Click on* ***Edit button****.*
- *Finally click on* ***Done button****.*

Delete a Folder style.

- *Click on* ***Delete a folder style icon****.*
- *Click on* ***Delelete button****.*

When a Folder style is deleted, all the folders that had it assigned will be reassigned to default Folder style.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create support zip

This feature allow the user to download a zip file with important info about current installation. It contains the following files:

- ***env.properties**: environment properties*
- ***openkm.log**: application log*
- ***openkm.properties**: OpenKM configuration file*
- ***openkm.xml**: OpenKM configuration file*
- ***server.xml**: tomcat configuration file*
- ***system.properties**: system properties*

This feature will download a zip file to your computer.

It will only download application log for the current day. Old logs will not be donwloaded.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Hibernate stats

With the Hibernate stats feature the administrator can take control of value information about database queries, finding out performance problems and the possible causes.

By default Hibernate stats feature is disabled because the log analysis consumes hardware resources.

This feature should only be enabled by administrator in order to investigate possible performance problems with application.

Each time the application is restarted all the previous analysis is removed. It's important that wh he or she n the administrator enables this feature, takes in consideration that the application should not be stopped.

Information provided by Hibernate stats

Information provided by Hibernate stats are shown in several tables:

- *General data.*
- *Query statistics.*
- *Entity statistics.*
- *Collection statistics.*
- *2nd level cache statistics.*

Time columns are finished with dur. name:

- *Max dur (Maximum duration).*
- *Min dur (MInimum duration).*
- *Avr dur (Average duration).*
- *Total dur (Accumulated total time).*

Columns with time values are in miliseconds.

When the application presents some performance problem, in most cases it will be caused by some query that need a huge time to complete its execution.

*On Query statistics are shown, all the database queries executed by application, the most important column value is **Max dur.** which indicates the Maximum duration of the query.*

Enable Hibernate stats

This option is only available when Hibernate stats is stopped.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *On right top click on?* *Enable button.*

Clear Hibernate stats

- *On right top click on?* *Clear button.*

Disable Hibernate stats

> *This option is only available when Hibernate stats is running.*

- *On right top click on?* *Disable button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

LDAP Search

The LDAP Search feature helps administrator to checking LDAP configuration.

> *Application LDAP integration is based on a subset of LDAP queries, for example retrieving LDAP users or LDAP groups among others.*
>
> *More information at LDAP configuration.*

These are the available parameters:

Field	*Description*
Host	*The LDAP host.*
User	*The LDAP user distinguised name.*
Password	*The LDAP user password.*
Search base	*The LDAP search base.*
Search filter	*The filter used in LDAP search.*
Attribute	*The atrribute returned by LDAP search.*
Referral	*In almost all cases this value is empty. Use "**follow**" to indicate several domain servers working together (balanced).*

Example of a query:

Field	*Description*
Host	*ldap://192.168.1.34:389*
User	*CN=Administrator,OU=openkm,DC=company,DC=com*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Password	**secret**
Search base	*DC=company,DC=com*
Search filter	*objectclass=user*
Attribute	*sAMAccountName*
Referral	*In almost all cases this value is empty. Use "**follow**" to indicate several domain servers working together (balanced).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

List indexes

The List index feature helps the administrator to take control about how the objects (documents, folders, mails and records) are indexed into Lucene search engine.

At the main screen is shown a table with the Lucene indexed fields of each object.

Common indexed fields:

Field	*Description*
#	*Number*
_hibernate_class	*The hibernate class name:* • *com.openkm.dao.bean.NodeFolder* • *com.openkm.dao.bean.NodeDocument* • *com.openkm.dao.bean.NodeMail* • *com.openkm.dao.bean.NodeRecord*
uuid	*Object unique identifier.*
parent	*Parent node.*
context	*Context values:* • *okm_root* • *okm_metadata* • *okm_categories* • *okm_thesaurus* • *okm_templates* • *okm_personal* • *okm_mail* • *okm_personal* • *okm_trash*
tenant	*Tenant identifier.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

author	*Id of the user author.*
created	*Indicate when object was created.*
name	*Object name.*
userPermission	*User permissions.*
rolePermission	*Role permissions.*

*Depending the type of object (**_hibernate_class**) can be more indexed fields.*

Search indexes

- *On top right click on* ***Search indexes button***.
- *In input box copy some object UUID (from desktop view, you can get at Properties tab any object UUID, select the UUID and press CTRL+C to copy).*
- *Click on* ***Search button***.
- *From the table list click on* ***List icon***.

*The filtering input box use the sintaxis **FIELD : value** , here some examples:*

- *uuid:ce3338a5-b1ec-481b-9d59-f5aeb3fcf42b*
- *name:invoice.**
- *text:telephone*

Let's see an example. Imagine you have two documents called "Parana.doc" and "Parana.pdf". You can locate them by this search:

```
name:paran*
```

But be careful because using quotation marks you won't get any result:

```
name:"paran*"
```

If you want to locate all PDF documents which name starts with "Paran", the right query is:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
name:paran*pdf
```

This query would also work (note the dot before que extension):

```
name:paran*.pdf
```

*But this query will failt because you cannot use a * or ? symbol as the first character of a search. This is a Lucene limitation.*

```
name:*pdf
```

If you want to lookup documents with a given extension, I would recommend using the MIME Type property this way:

```
mimeType:application/pdf
```

In this query we are looking for all documents which name starts with "parana" and has a keyword called "dog":

```
name:parana* AND keyword:song
```

You can learn more about query syntaxt at Lucene Query Syntax.

Enable show terms

When show terms enabled, is shown and extra fields named terms on documents objects with the terms used by Lucene search engine for indexing that document.

- *On top right click on* ***Show terms button***.

Disable show terms

- *On top right click on* ***Show terms button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

LogCat

The LogCat feature helps administrator to accessing Tomcat log files directly from application.

Logs are shown is a table structure:

File	Action

*The main Tomcat log file is named **catalina.log**.*

This file rotate each day at night creating a new catalina.log file and saving older with name catalina.log.year-month-day

Purge all logs

The purge option will clean all Tomcat logs.

- *On top right click on* ***Purge button***.

Download file as ZIP

- *Choose a log file.*
- *Click on* ***download as ZIP icon***.

Browser log file

- *Choose a log file.*
- *Click on* ***browse icon***.
- *Use filter options.*

Filter options:

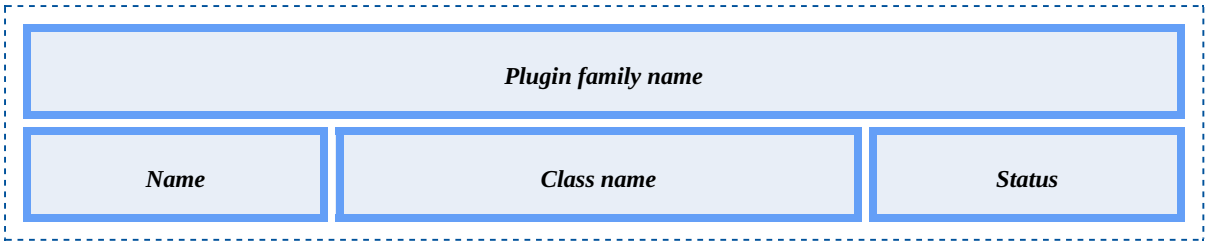
Option	Description
Begin	*Indicate the row position.* *By default 0 indicate the starting of the document.*
End	*Indicate the row position.* *By default -1 indicate the end of the document.*

String	*Filter by some text.*

Plugins

The application comes with a plugins feature system that offers an easy way to customize and extend OpenKM functionalities. By default application comes with general purpose plugins implementation, these can be disabled, changed by others or increased in number depending the type of the plugin. For example in antivirus plugin has sense disable or replace by other, on case text extractor has sense disable or increase the number of available text extractors.

Plugins are shown in a table structure:



Plugin families:

Plugin family	*Module*	*Description*
Automation validations	*Automation*	*Validations are used in automation feature.* *More information at:* • *Automation* • *Creating your own Automation Validation*
Automation actions	*Automation*	*Validations are used in automation feature.* *More information at:* • *Automation* • *Creating your own Automation Action*
Template parsers	*OCR Zone*	*Data extraction parser used in OCR Zone.* *More information at:* • *OCR templates* • *Creating your own OCR Data Capture plugin*

Template control parsers	*OCR Zone*	*Document identification check point used in OCR Zone.* *More information at:* • *OCR templates* • *Creating your own OCR Validation Control plugin*
Option select values	*Metadata*	*Retrieve select values from server side logic. For example helps on retrieving data from third-party system.* *More information at:* • *Metadata* • *Metadata Select field* • *Creating your own Option Select Values plugin*
Field validators	*Metadata*	*Validation based on server side logic. Helps on implementing complex validation logics that are not covered from default UI validators.* *More information at:* • *Metadata* • *Metadata Validator element* • *Creating your own Form Validator plugin*
Principal adapters	*Authentication*	*Authentication adapters are used to retrieve users and roles information. For example connect to some LDAP, or external database that stores users and roles.* *More information at:* • *Creating your own Principal Adapter*
Version number adapters	*Document*	*Implements version number logic.* *More information at:* • *Creating your own Version Number Adapter*
Document	*Conversion*	*Conversion service is used to convert document to other formats.*

converters		*More information at:* • *Creating your own Document Converter*
Metadata adapters	*UI*	*Visualization logic for metadata UI.* *More information at:* • *Creating your own Stack Metadata Adapter*
Text extractor	*Search engine*	*Text extractor based on document MIME TYPE.* *More information at:* • *Creating your own Text extractor*
Antivirus	*Document*	*Antivirus checker. Helps on integrating with any command line antivirus.* *More information at:* • *Creating your own Antivirus Analyzer plugin*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profiling stats

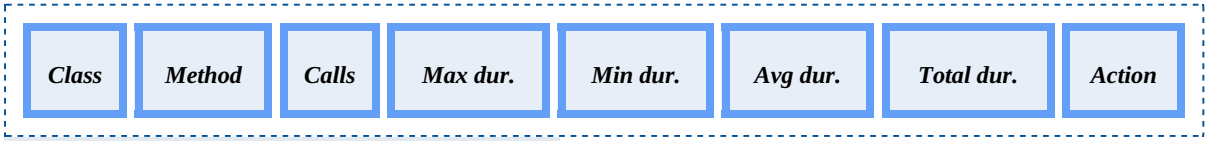
The Profiling stats feature gives low level value information to the administrator to find out application performance problems.

By default the Profiling stats feature is disabled because the log analysis consumes hardware resources.

This feature should only be enabled by the administrator in order to investigate possible performance problems with application.

Each time the application is restarted all the previous analysis are removed. It's important when administrator enable this feature take into consideration that the application should not be stopped.

Profiling stats are shown is a table structure:



Field	*Description*
Class	*The Java class name.*
Method	*The Java method name into the class.*
Calls	*Number of calls.*
Max dur.	*Maximum duration of the calls.*
Min dur.	*Minimum duration of the calls.*
Avr dur.	*Average duration of the calls.*
Total dur.	*Accumulated duration time.*
Action	*List of all calls.*

The duration time is in miliseconds.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Enable profiling stats

- *On top right click on* ***Enable button***.

Disable profiling stats

- *On top right click on* ***Disable button***.

Clear profiling stats

- *On top right click on* ***Clear button***.

Refresh profiling stats

- *On top right click on* ***Refresh button***.

List of detailed calls

- *Click on* ***List detailed calls icon***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Purge permissions

The purge permissions feature helps administrators to delete users and roles grants assigned to objects (documents, folders, mails or records).

Examples of usage:

- *Remove grants of users or roles that no longer exists into the repository.*
- *Remove grants of users or roles that has been propagated along the repository accidentally.*

Purge

This operation can not be reverted, use with care.

- *From checkbox **choose users or roles** option.*
- *Click on **List Button**.*
- *Click on **Delete icon**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Purge trashes

The purge trashes feature helps the administrator to clean all users trashes.

When a users deletes objects (folders, documents, mails or records) they go into users trash to prevent accidentally deletes and be able to recover lost data.

Although the users have a feature to clean their own trash, usually they do not do it. That could cause to having a lot of garbage into the trashes.

The administrator periodically should check the repository growing an specially the size of the trashesthey .

Purge

This operation can not be reverted. Use with care.

- ***Write Yes** into the input box.*
- *Click on **Purge button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Rebuild indexes

The Rebuild Indexes feature helps the administrator to rebuild Lucene indexes.

Indexes	*Description*
Text extractor	*Extracts all document contents and rebuild the indexes.*
Lucene indexes	*Rebuild indexes with actual contents.* *When you choose this option you can rebuild all node types or only the selected. Available options are:* • *NodeDocument for rebuilding nodes of type document.* • *NodeFolder for rebuilding nodes of type folder.* • *NodeMail for rebuilding nodes of type mail.* • *NodeRecord for rebuilding nodes of type record.*
Optimize indexes	*Remove older contents that no longer exists in repository.*

Rebuild

- *Choose the index to rebuild.*
- *Click on **Execute button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Repository checker

The repository checker feature helps the administrator to verify the repository integrity.

When the administrator thinks the system could be compromised, for example a hard disk failure or similar,he can use Repository checker feature to verify the application integrity.

Option	Description
Path	*The path from where will be started the verification process.* *Common values:* • */okm:root* • */okm:trash* • */okm:personal* • */okm:mail* • */okm:templates*
Fast	*The fast verification will only check the existence of the last document version.*
Versions	*Verify the existence of all document versions. This option has priority over fast option.*
Checksum	*Calculate the checksum of repository document and compares with database value.*

Check repository

The process can take several minutes or hours depending on the repository size and the repository options checked.

The repository checker process will show a process status in %, before it is shown the application calculates the total amount of nodes that should be checked in a huge repository this calculation process can take several minutes, please be patient.

- *Set the path.*
- *Choose repository checker options (fast, versions, checksums).*
- *Click on* ***Check button****.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Repository export

The repository export feature helps the administrator to export the whole or part of the OpenKM repository to the file system.

The output result will be the same folders and files structure that are set into the application.

Field	Description
Repository path	*The repository path to export.* *Common values:* • */okm:root* • */okm:templates* • */okm:personal* • */okm:mail*

Filesystem path	*Server destination path.*
	Linux example:
	`/mnt/export`
	Windows example:
	`c:/export`

Metadata	*When enabled, metadata values are exported.*
History	*When enabled, all documents versions are exported.*

Export

- *Set the repository path.*
- *Check the repository export options.*
- *Click on Export button.*

Understanding output format

Case metadata and history are not checked

The result will be a folder structure with documents and mails into. Although a document has several versions only will be

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

exported the actual version.

Case only metadata is checked

The result will be a folder structure with documents and mails into. Although a document has several versions only will be exported the actual version.

For each folder, document or mail will be created an ***.okm*** *file. For example for a file named "Administration.html" will be created in the same path a file named "Administration.html.okm". The* ***.okm*** *files contains into all the metadata information associated to the node in a JSON format.*

Example of .okm file:


```
{"uuid":"8299d059-7894-468b-ba48-3f3b59ebc37c",
 "author":"okmAdmin",
 "name":"redacted_481.pdf",
 "path":"/okm:root/redacted_481.pdf",
 "created":{"year":2015,
   "month":6,
   "dayOfMonth":1,
   "hourOfDay":17,
   "minute":18,
   "second":0},
   "lastModified":{"year":2015,
       "month":6,
       "dayOfMonth":2,
       "hourOfDay":10,
       "minute":47,
       "second":39},
   "language":"en",
   "keywords":[],
   "categories":[],
   "subscriptors":[],
   "notes":[{"user":"system",
       "date":{"year":2015,
         "month":6,
         "dayOfMonth":2,
         "hourOfDay":10,
         "minute":47,
         "second":40},
       "text":"New version 1.1 by okmAdmin: Signed",
       "path":"7d11d393-df46-49a1-8f29-ae6d0ea4a093"}],
   "propertyGroups":[],
   "version":{"author":"okmAdmin",
        "name":"1.1",
        "created":{"year":2015,
          "month":6,
          "dayOfMonth":2,
          "hourOfDay":10,
          "minute":47,
          "second":39},
        "size":139215,
        "mimeType":"application/pdf",
        "comment":"Signed"},
 "grantedUsers":{"okmAdmin":15},
 "grantedRoles":{"ROLE_USER":15}
}
```


Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Take in consideration that the JSON output format can change if the main objects (folder, document, mail or record) are added, removed or modified fields.

Case only history is checked

The result will be a folder structure with documents and mails. All document versions will be saved.

Example of output for a document with version 1.0 and 1.1:

- *Administration.html (size = 0 bytes)*
- *Administration.html#v1.0# (contents of version 1.0)*
- *Administration.html#v1.1# (contents of version 1.1)*

Case metadata and history are checked

The result will be a folder structure with documents and mails. All document versions will be saved.

For each folder, document or mail will be created a ***.okm*** *file. For example for a file named "Administration.html" will be created in the same path a file named "Administration.html.okm". The* ***.okm*** *file contains all the metadata information associated to the node in a JSON format.*

Example of output for a document with version 1.0 and 1.1:

- *Administration.html (size = 0 bytes)*
- *Administration.html.okm (metadata associated to the document)*
- *Administration.html#v1.0# (contents of version 1.0)*
- *Administration.html#v1.0#.okm (metadata associated to version 1.0)*
- *Administration.html#v1.1# (contents of version 1.1)*
- *Administration.html#v1.1#.okm (metadata associated to version 1.1)*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Repository import

The repository importer helps administrator to import data from the file system.

Initial import (import standard folders and documents from your openkm server).

Import from previous openkm export.

Field	Description
Filesystem path	*Server import path.*
Repository path	*The repository destination path.* *Common values:* • */okm:root* • */okm:templates* • */okm:personal* • */okm:mail*
Metadata	*When enabled, the importer assumes the metadata information (.okm files) are present.*
History	*When enabled, the importer assumes all document versions of the document are present.* *More information about document versions format at Repository export.*
Restore UUIDs	*When enabled, the importer will keep the same node UUID.* *When applicationis integrated with thirdparty software, usually the thirdparty software stores in their own database UUID of the node as main reference. This references should not be lost, that's why on these scenarios are needed to keep out the UUID's.*

Specially when importing from previous openkm, export must take in consideration that you must use the same import options that had been chosen while exporting, otherwise the import will fails.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Import

- *Set the file system path.*
- *Set the repository path.*
- *Check the repository import options.*
- *Click on **Import button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Scripting

Scripting was an advanced feature that enables administrators to execute some BeanShell scripts in folders, triggered on every notified event (for example uploading documents).

This feature has been enhanced by Automation (see ExecuteScripting at Built-in Automation Action plugins), which helps you to do the same things with a few mouse clicks. For compatibility reasons there is also an Automation task where you can put a BeanShell code.

OpenKM uses BeanShell. For more information check http://www.beanshell.org/intro.html.

BeanShell is a small, free, embeddable Java source interpreter with object scripting language features, written in Java. BeanShell dynamically executes standard Java syntax and extends it with common scripting conveniences such as loose types, commands, and method closures like those in Perl and JavaScript.

See some examples at Scripting samples.

Execute script as part of application starting process

Can be executed as an script as a part of application starting process. To do so, just create a ***start.bsh*** *file in* ***$TOMCAT_HOME****.*

For example, we use this feature to create a complete environment (create custom users, register property groups, register workflow) each time the OpenKM demo is restarted.

Execute script as part of application stopping process

Can be executed a an script as part of the application stopping process. To do so, just create a ***stop.bsh*** *file in* ***$TOMCAT_HOME****.*

Scripting in Automation

In the Automation Scripting execution, there're several variables that are directly injected in scripting environment that can be used.

List of variables what can be used directly:

Variable	Description
systemToken	*The system token.*

node	*Involved node (NodeDocument, NodeFolder, etc.)*
uuid	*Node unique identifier.*
file	*Only available when uploading or updating a new document content.*
userId	*The user id.*

Example:

```
import com.openkm.api.*;

// Variable uuid is injected by automation to the scripting environment
OKMProperty.getInstance().addKeyword(null, uuid, "test");
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

System properties

The system properties feature show the JVM variables.

Most common JVM variables:

Key	Description of Associated Value
`java.version`	Java Runtime Environment version
`java.vendor`	Java Runtime Environment vendor
`java.vendor.url`	Java vendor URL
`java.home`	Java installation directory
`java.vm.specification.version`	Java Virtual Machine specification version
`java.vm.specification.vendor`	Java Virtual Machine specification vendor
`java.vm.specification.name`	Java Virtual Machine specification name
`java.vm.version`	Java Virtual Machine implementation version
`java.vm.vendor`	Java Virtual Machine implementation vendor
`java.vm.name`	Java Virtual Machine implementation name
`java.specification.version`	Java Runtime Environment specification version
`java.specification.vendor`	Java Runtime Environment specification vendor
`java.specification.name`	Java Runtime Environment specification name
`java.class.version`	Java class format version number
`java.class.path`	Java class path
`java.library.path`	List of paths to search when loading libraries
`java.io.tmpdir`	Default temp file path
`java.compiler`	Name of JIT compiler to use

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

`java.ext.dirs`	*Path of extension directory or directories*
`os.name`	*Operating system name*
`os.arch`	*Operating system architecture*
`os.version`	*Operating system version*
`file.separator`	*File separator ("/" on UNIX)*
`path.separator`	*Path separator (":" on UNIX)*
`line.separator`	*Line separator ("\n" on UNIX)*
`user.name`	*User's account name*
`user.home`	*User's home directory*
`user.dir`	*User's current working directory*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Thesaurus

The thesaurus feature helps the administrator to create the thesaurus folder structure. By default the thesaurus folder structure is not created.

The application needs a configured parameter to create thesaurus folders, check Thesaurus configuration.

Take a look a the full configuration example at Creating your own thesaurus.

Build thesaurus

- *Choose the level (the level indicates the tree depth that will be shown during creation process).*
- *On top right click on* *Generate button.*

A log will be displayed on the screen to monitor thesaurus folder creation. This process could take some hours depending on your thesaurus size.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

User guide

You can use the Quick start guide to quickly learn the basics, and to find detailed info on the user sections.

OpenKM frontend is the Back office view on the content repository. The features are organized in several sections:

- *The **Menu bar** at the top where are available all the features organized by type.*
- *The **Quick search box** at the right top.*
- *The **Toolbar** where are the most commonly used features.*
- *The **Desktop** tab enables to navigate the repository.*
- *The **Search** tab enables to query the repository.*
- *The **Dashboard** tab provides a set of dashboards.*
- *The **Administration** tab provides a set of the administration tools.*
- *The **Status** panel at the bottom provides general information.*

*Only members with "**ROLE_ADMIN**" role have access to the **Administration** tab.*

The OpenKM frontend allows you to manage any type of content: scanned images, photographs, office documents and even large videos files or CAD files among others. The built-in workflow feature helps companies to automate businesses processes and get a full control of the cycle of the documents, saving time and money. It also allows the social activities around the content to be used to connect people to other people, information to information, and people to information; helping to manage, more efficiently, the collective intelligence of the human resources of the company.

The OpenKM Profiles feature helps administrators to set what is displayed for each user of the application.

OpenKM provides the solution to:

- *Controls the growing volume of information to be managed.*
- *Ends the diversification of information storage sources.*
- *Prevents the duplication of the information.*
- *Obtains valid information through the control of document versions.*
- *Reduces the time and resources used in locating information.*
- *Avoids knowledge leaks or wrongful access by users, therefore complying with the obligations set in the LOPD (Spanish Data Protection Law).*
- *Restricts non-authorized users access to the information.*

Through the OpenKM system, all relevant and useful information can be accessed and found in the less possible time to resolve problems and make decisions.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Menu bar

The menu bar at the top has all features organized by type:

- *File.*
- *Edit.*
- *Tools.*
- *Bookmarks.*
- *Templates.*
- *Helps.*

Quick search box

The Quick search box is at the top right corner. It provides a fast access to Default search in Search panel.

The minimal text length for doing a query is three characters.

ToolBar

The toolbar section has the most commonly used features like download documents, create new folders, edit or upload files among others.

Desktop

Most user's activity is done there. Users can work directly on folders, documents, mails or records.

The desktop is distributed in tree sections:

- ***Navigation** panel at left hand side.*
- ***Browser** panel at top center.*
- ***Properties** panel at bottom center.*

Navigation

The navigation at left hand side is an stack panel. Each stack contains into a tree (folder and records hierarchy) structure

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

from root that enables easily access documents and mails.

Browser

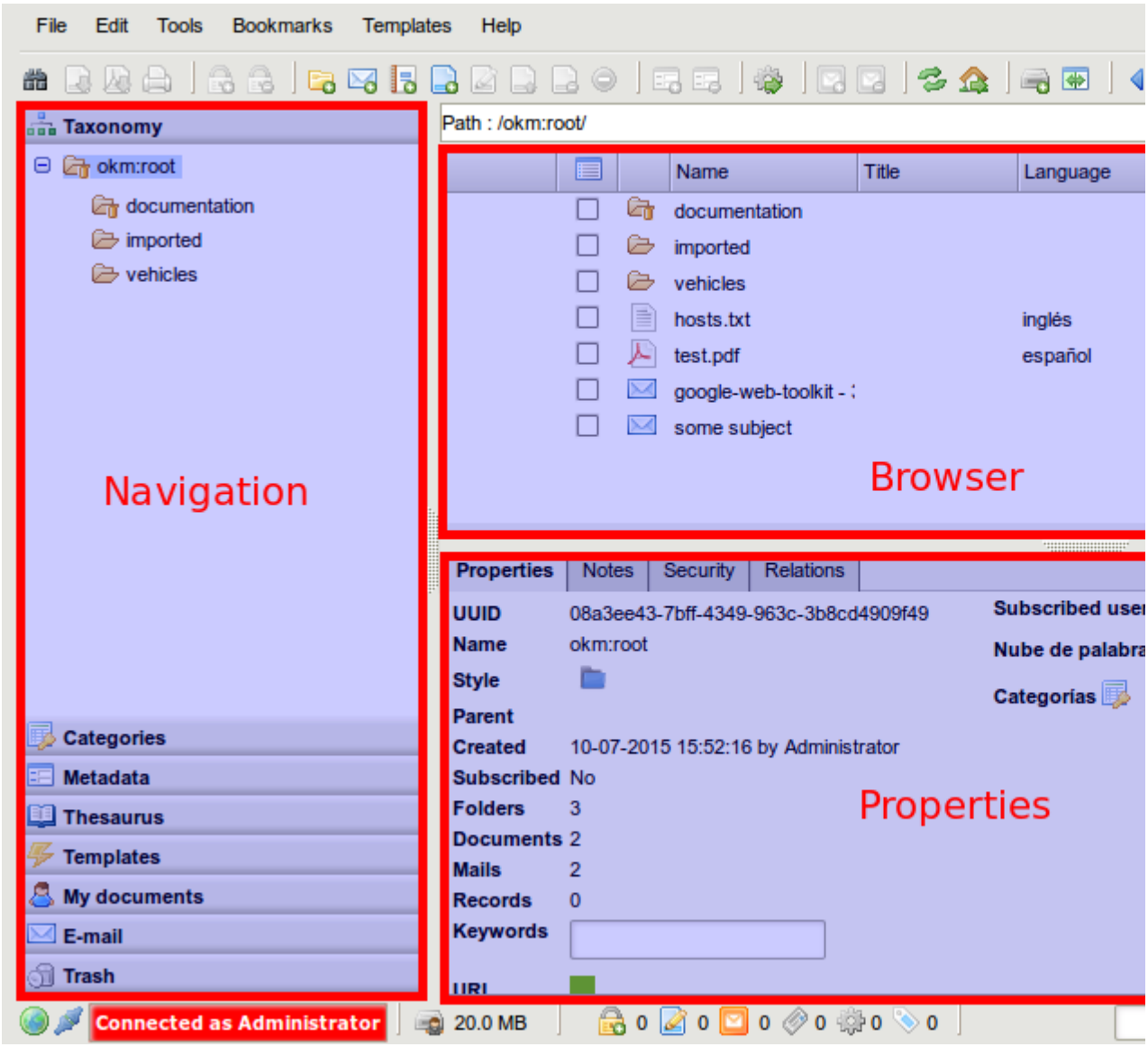
The browser panel at top center shows a table of nodes (documents, folders, mails and records) that are on the selected node of navigation panel.

Properties

The properties panel at bottom center shows a collection of tabs that contain the grouped properties or features like preview associated with selected node. The selected node can be a navigator node or browser node.

Basically the selected node at navigator panel determines the content shown at the browser panel and depending if the last selected node by the user is at navigator or browser panel the properties panel will shown the properties of the active node at navigator or browser.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

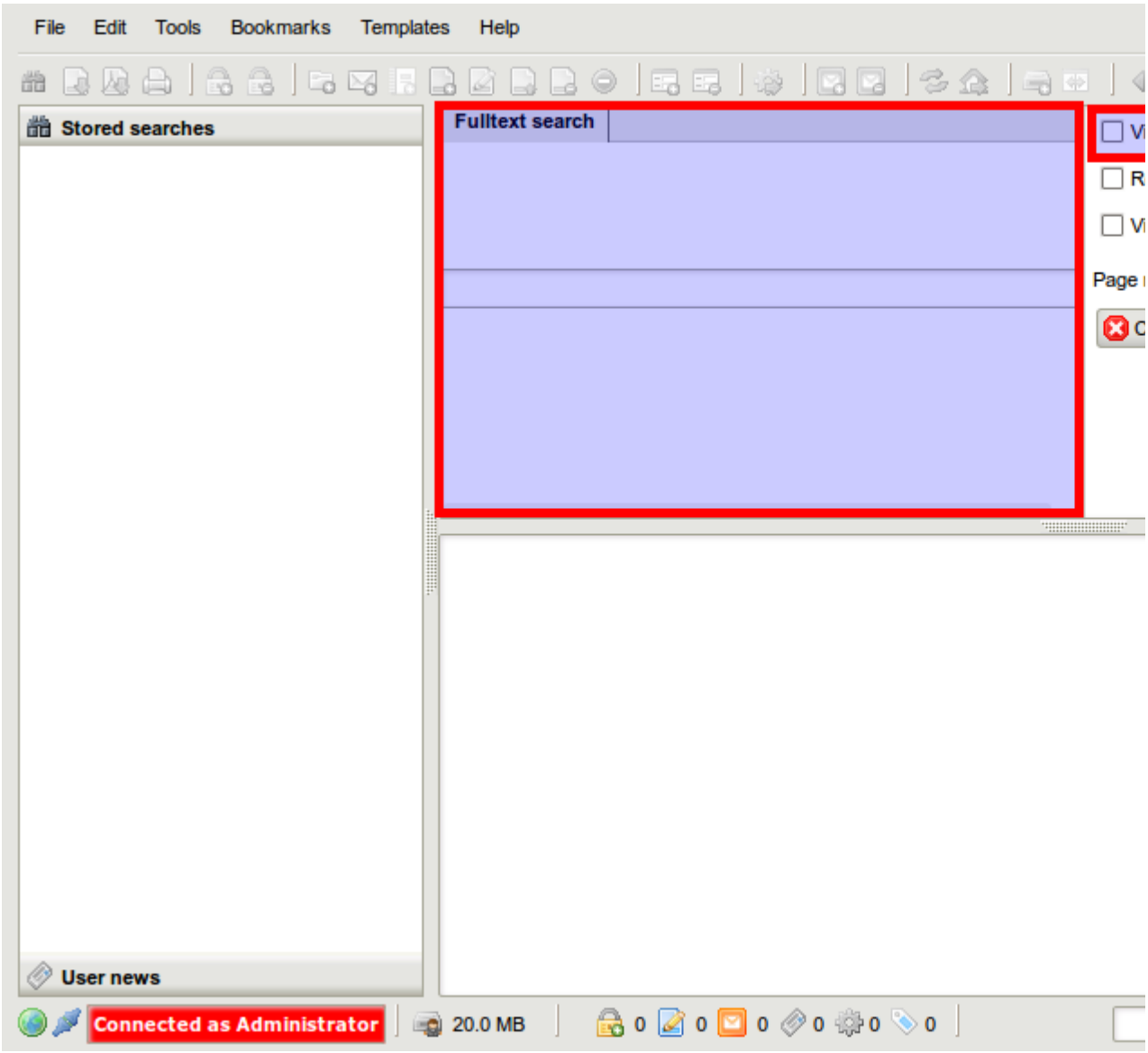


Search

The search tab enables to query the repository. This tab gives access to different searches to query the content of the application:

- ***Default search** is a full-text search. Default search is also available from the top right corner on the screen.*
- ***Advanced search** enables you to search keywords, metadata and additional criteria such as location in the application.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Dashboard

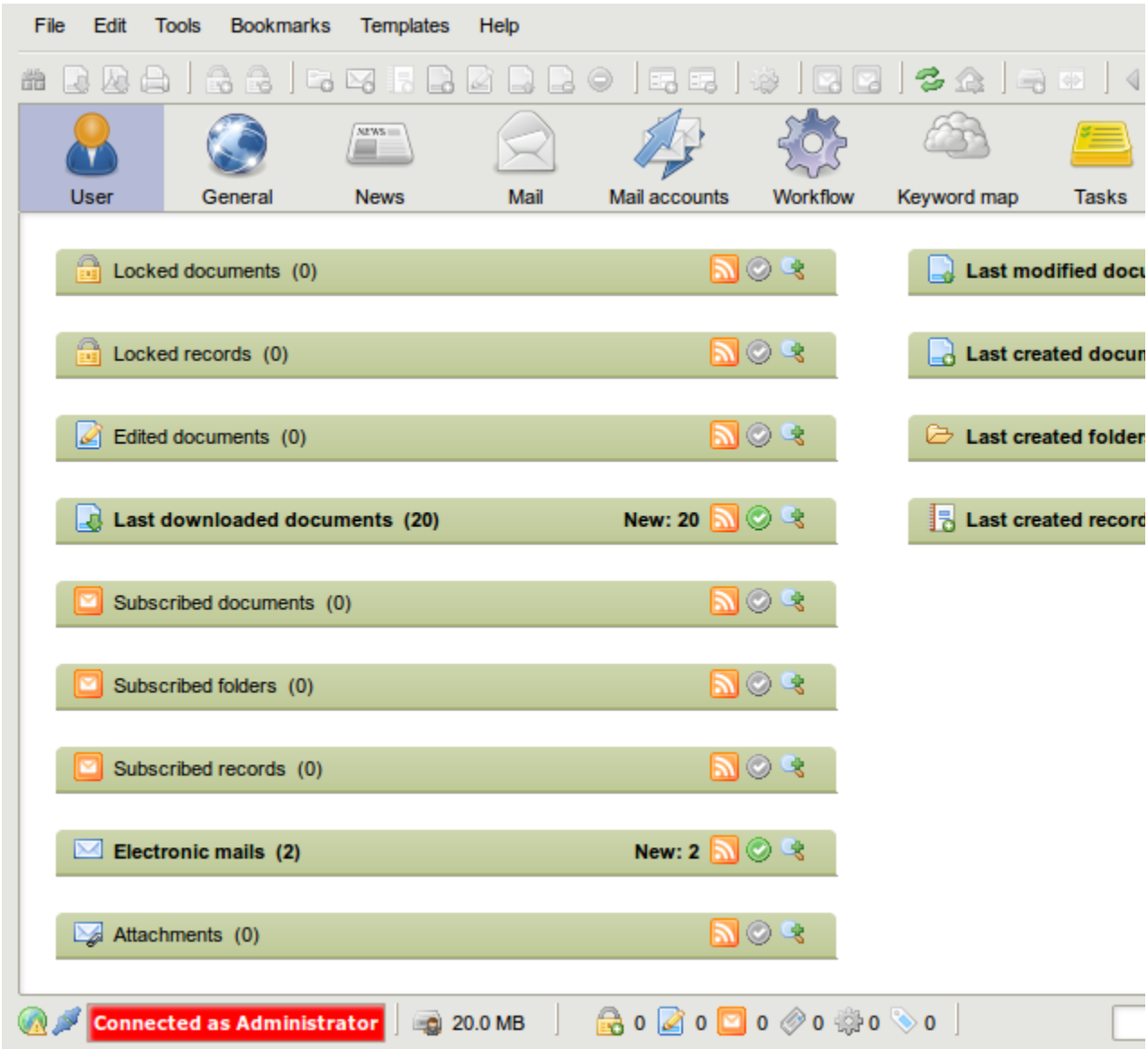
The dashboard panel provides a set of dashboards, each user has a set of personal dashboard that display the information relevant for him and helps to have a global view of the application activity.

The dashboard panel :

- ***User view**. Displays the actions that have been carried out in the system*
- ***General view**. Displays a global view of the application activity.*
- ***News**. Displays information based on scheduled user search queries.*
- ***Mail**. It's a virtual navigation across all mail repositories.*
- ***Mail account**. Helps users to managing their own mail accounts.*
- ***Workflow**. Users work in company business process tasks.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Keyword map***. *Displays a tag cloud based on all repository keywords.*
- ***Tasks***. *Helps on creating, administrating, assigning and following basic company tasks.*
- ***Calendar***. *Helps on following the status of the basic company task.*
- ***Messaging***. *Displays internal OpenKM messaging.*
- ***Forum***. *Displays a forum.*
- ***Wiki***. *Displays a wiki.*



Administration

The administration panel is a set of tools that helps administrator to manage the application.

> *More information at Administration guide.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Status

The status panel provides general information:

- *OpenKM news alert.*
- *The connected user.*
- *User repository size.*
- *Chat.*
- *Alerts.*
- *HTML documents on edition.*
- *Message panel.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Quick start

Before starting a production environment you should also read these sections:

- *Essential fundamentals.*
- *Plan your data organization.*

With OpenKM it's simply to store and share information and collaborate effectively.

This quick start guide give you an introduction to some of the features of the OpenKM:

- *Logging in.*
- *Create a taxonomy.*
- *Add documents.*
- *Edit documents.*
- *Search documents.*

Loggin in

You need to log in to OpenKM.

- *Enter the following on the login page:*
  - ***Username**: your username.*
  - ***Password**: your password.*
- *Choose your **language**.*
- *Click on **Login** button.*

Ask your administrator for your login URL. By default from the OpenKM server the URL is http://127.0.0.1:8080/OpenKM

The default administrator credentials are:

- ***Username**: okmAdmin*
- ***Password**: admin*

*Do not forget to **change the administrator password**.*

Create a taxonomy

The dictionary says:

```
"Taxonomy (from Greek taxis meaning arrangement or division and nomos
```

```
meaning law) is the science of classification according to a pre-
determined system, with the resulting catalog used to provide a
conceptual framework for discussion, analysis, or information retrieval.
In theory, the development of a good taxonomy takes into account the
importance of separating elements of a group (taxon) into subgroups
(taxa) that are mutually exclusive, unambiguous, and taken together,
include all possibilities."
```

*Basically, a **taxonomy** is a way to group things together. Like a controlled vocabulary focuses on concepts, i n practice, a good taxonomy should be simple, easy to remember, and easy to use.*

The first step should be analyse the company organization and define a set of folder hierarchy to store the documents organized.

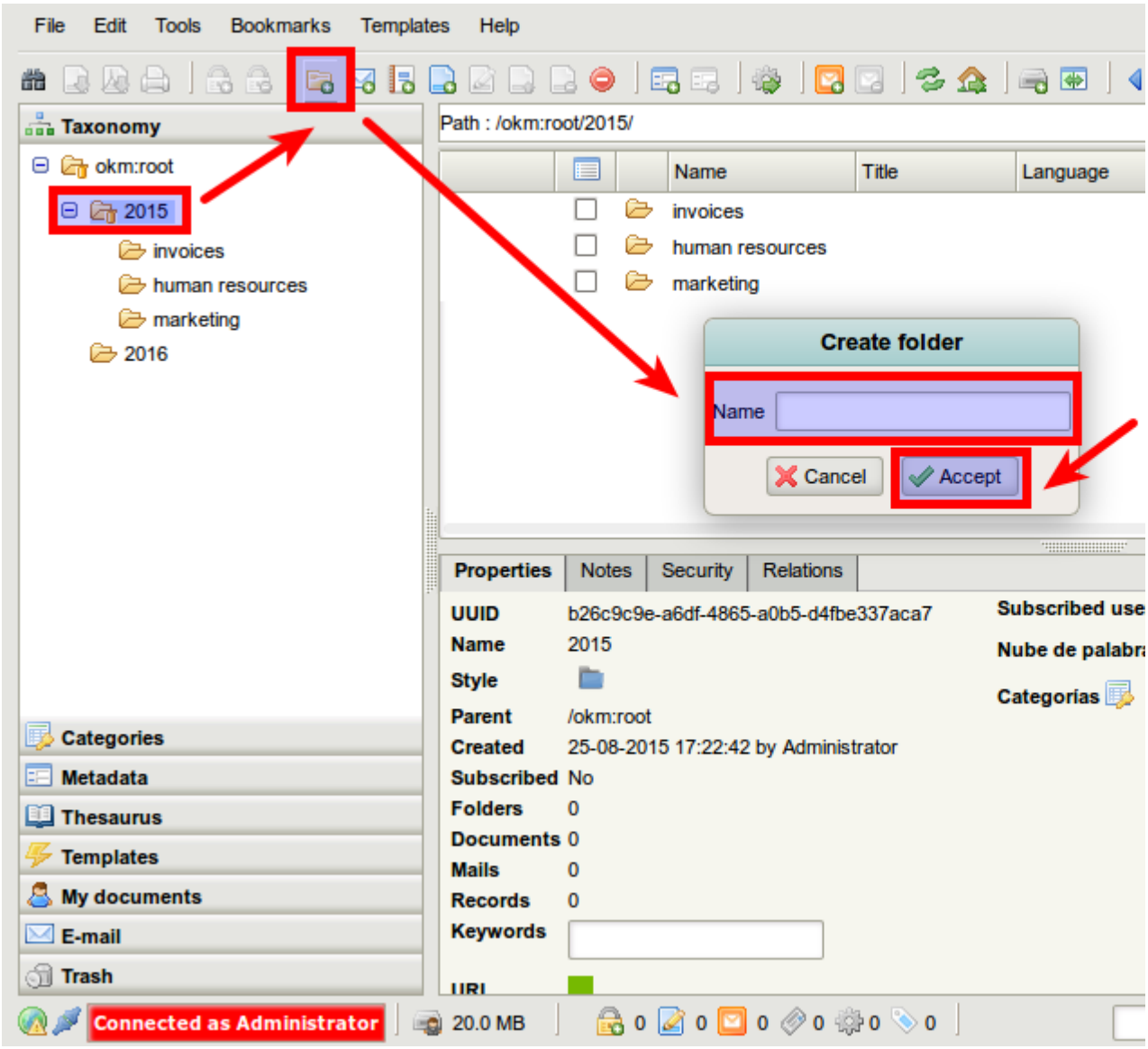
Take this one as a sample:

/okm:root/	***2015/***	***invoices/***
		human resources/
		marketing/
	2016/	***invoices/***
		human resources/
		marketing/

To create a folder:

- ***Select a node** from taxonomy where to create the folder.*
- *Click on* ***folder icon** at tool bar.*
- *Set the **name**.*
- *Click on **Accept** button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

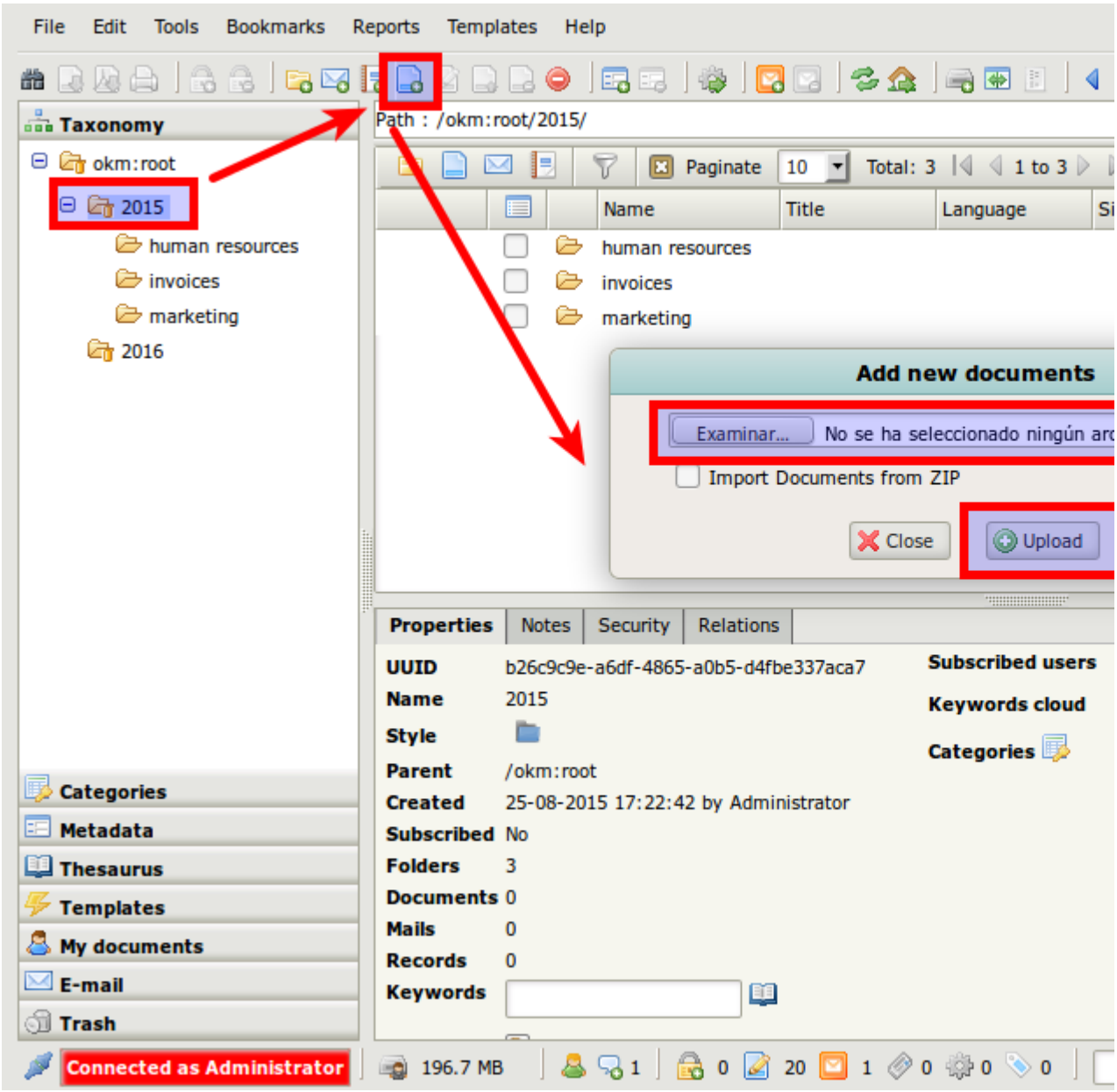


> *We suggest spending some time thinking and designing your taxonomy. For a best UI performance we suggest less than 200 elements per folder.*

Add documents

- ***Select a node*** *from taxonomy where to add the document.*
- *Click the* ***add document*** *icon from the toolbar.*
- ***Select the document*** *from your computer.*
- *Click* ***Upload*** *button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Edit documents

*The document edition is based on the "**check-out / check-in**" model.*

> *"**Check-out**" and "**check-in**" are common terms in many document management systems, including OpenKM. Their purpose is to prevent conflicts in an environment where multiple people might want to edit the same piece of content - in OpenKM a document - at the same time.*
>
> *The term "**check-out**" describes the process of getting a version of a document. By checking out a document, a user can prevent others from editing that content.*
>
> *The term "**check-in**" describes the process of updating a document to replace the previous version. By checking in the item, the user can allow others to edit the document, without needing to worry about overriding changes*

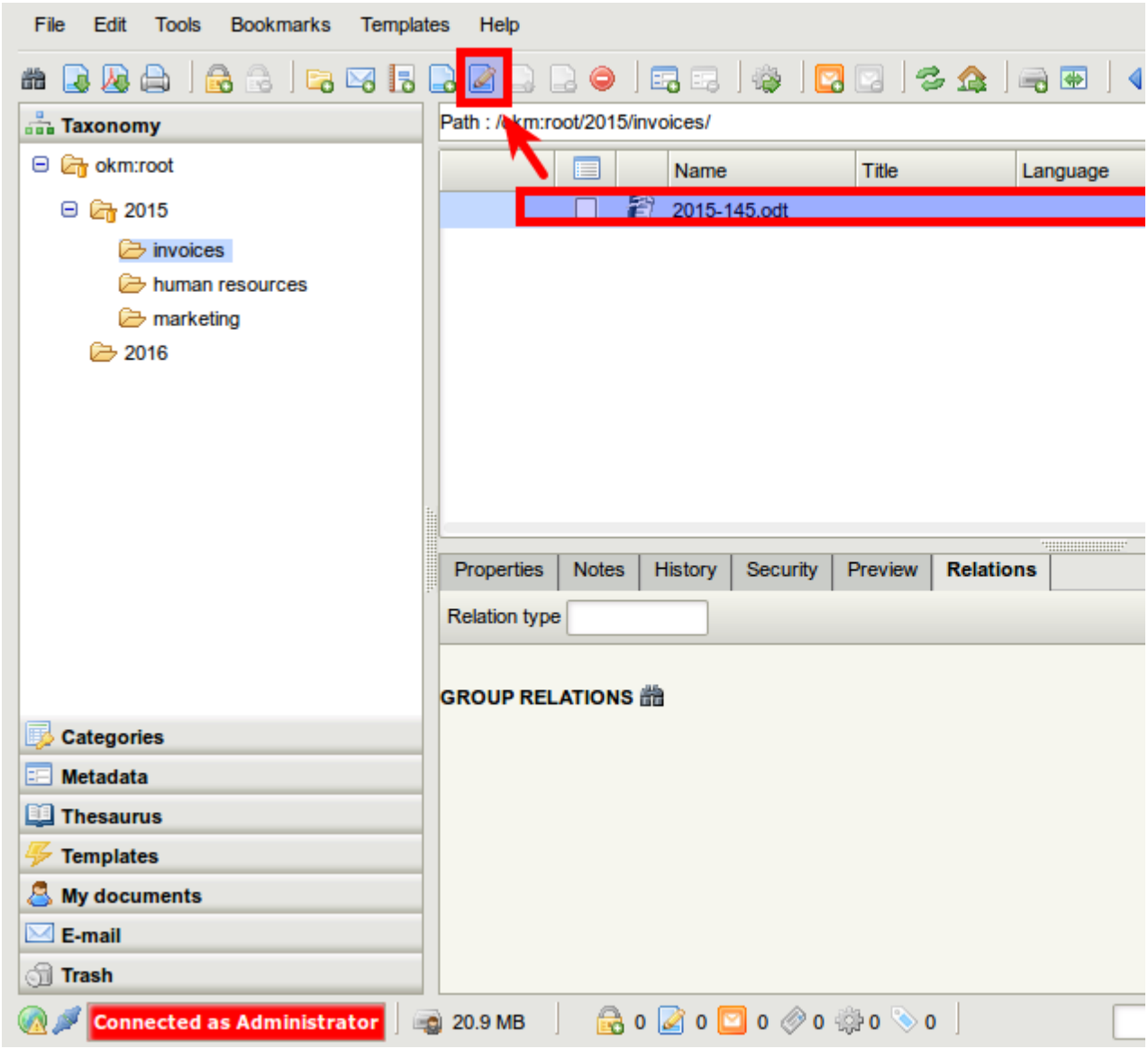
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

that others have made.

When a document is checked out to a user, that user can work on that document, and other users cannot. Other users must wait for the user to check in the document before they can edit. While the document is checked out, other users usually can see only the last version that was checked in; they can't see the changes that the current user has made on the document while it's been checked out.

First must check-out the document:

- ***Select a document***.
- *Click on* ***Edit icon***.

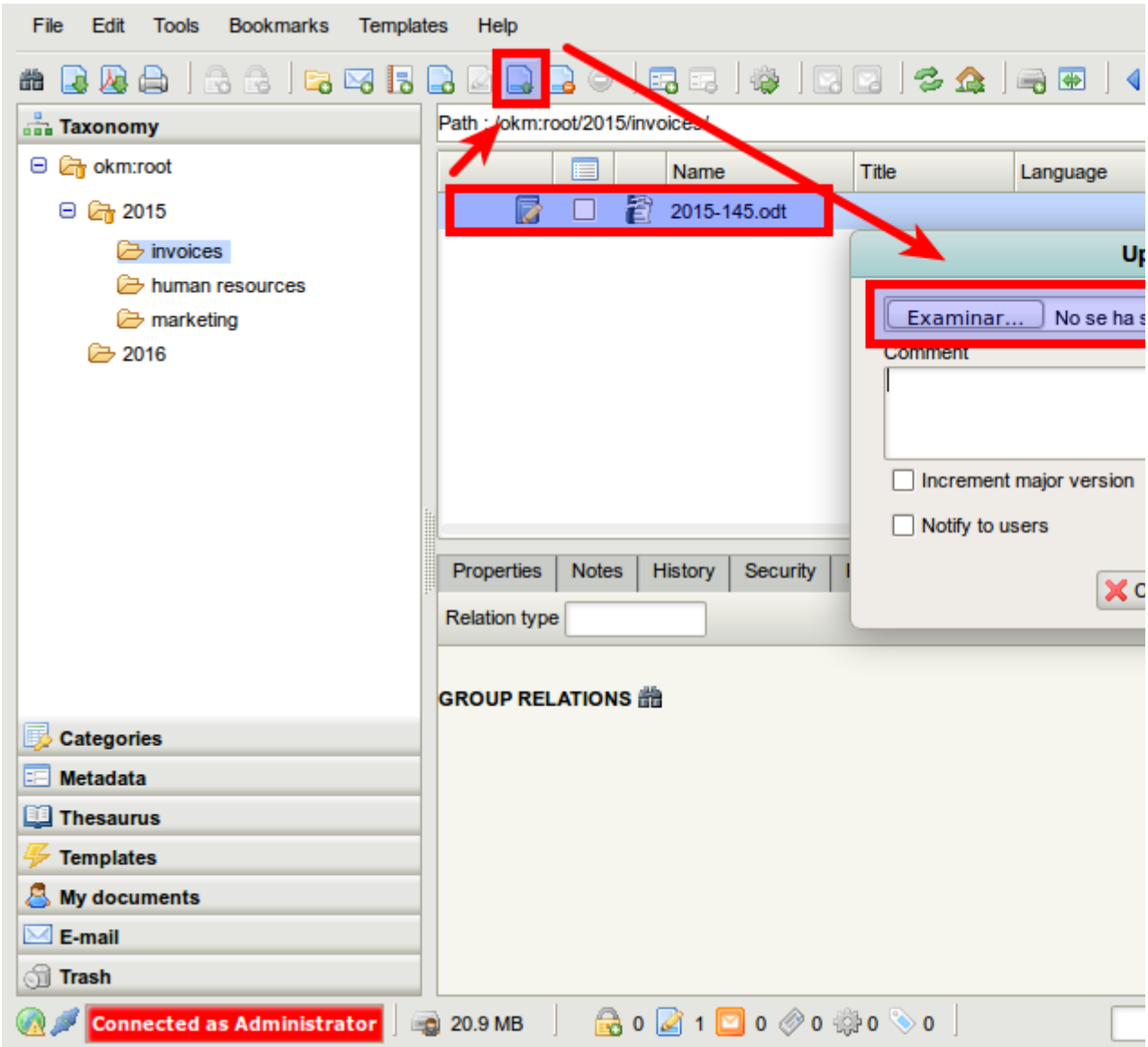


Finally check-in the document to upload the changes:

- *Select the* ***edited document***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

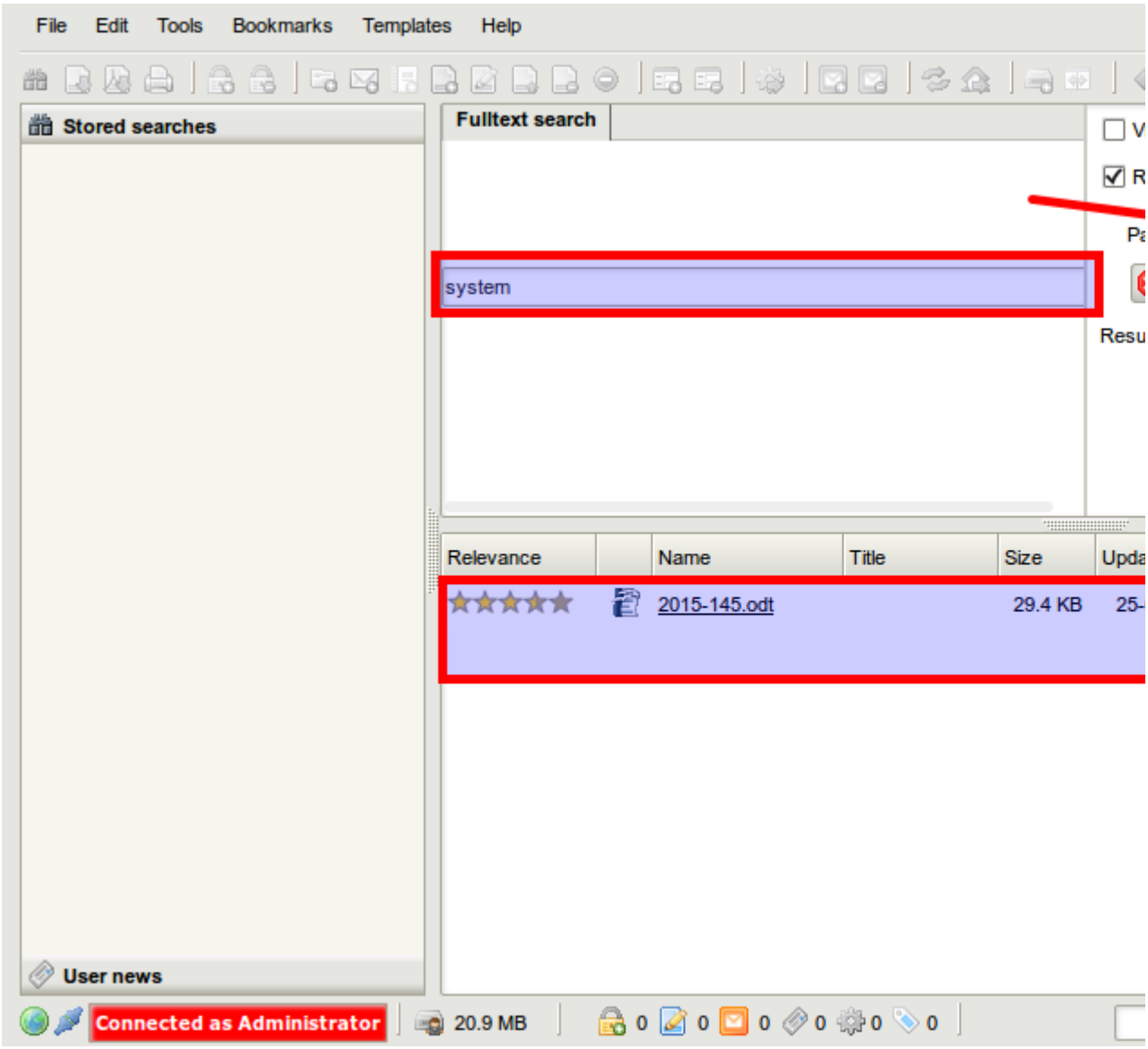
- *Click on* ***Update icon****.*
- ***Select the document*** *from your computer.*
- *Click* ***Upload*** *button.*



Search documents

- *Go to* ***search view****.*
- ***Write the term*** *for the query.*
- *Click on* ***Search*** *button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


File
Edit
Tools
Bookmarks
Templates
Help
Stored searches
Fulltext search
system
Relevance
Name
Title
Size
2015-145.odt
29.4 KB
User news
Connected as Administrator
20.9 MB

Essential fundamentals

Basic concepts

OpenKM has built-in with several types of nodes:

- *Folder.*
- *Document.*
- *Mail.*
- *Record.*

*A node type **folder** is a **storage unit or a container**. As a document, it also has a name but is not a fil, but it can contain documents, records and other folders into.*

*A node type **document** is a **file**. A document serves to establish one or several facts, and can be relied upon as a proof thereof.*

*A node type **mail**, most commonly referred to as **email** or **e-mail**, is a method of exchanging digital messages from an author to one or more recipients. A mail can contain documents (in mail context are named attachments).*

*In essence there are basically two kinds of nodes of type **record**:*

- *Non-digital records.*
- *Digital records.*

*A **non-digital record** is only a set of metadata groups.*

*A **digital record** can contains documents, mails, folders and other record into.*

> ***Non-digital record sample:***
>
> *When OpenKM is used to store the location of papers in the real world. Each record only is a set of metadata fields that helps user to locate papers and there's not binary information associated (like associated scanned document, etc.).*
>
> ***Digital record sample:***
>
> *OpenKM is used to store business records, where each record represents the result of some business process transaction. The digital record then, is used to store documents, mails associated with that process.*

Understanding taxonomy, keywords, categories and metadata groups.

Spending time on nodes categorization is a good practice. A good categorization will help users quickly finding fast of the desired information, using less time, saving money and being more efficient in your business process.

OpenKM helps you on it providing basically four ways for recategorizing your repository:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Taxonomy.*
- *Keywords.*
- *Categories.*
- *Metadata groups.*

You can use one of these features to categorize your repository or a combination of all.

Taxonomy

The dictionary says:

```
"Taxonomy (from Greek taxis meaning arrangement or division and nomos meaning law)
is the science of classification according to a pre-determined system, with the
resulting catalogue used to provide a conceptual framework for discussion,
analysis, or information retrieval. In theory, the development of a good taxonomy
takes into account the importance of separating elements of a group (taxon) into
subgroups (taxa) that are mutually exclusive, unambiguous, and taken together,
include all possibilities."
```

*Basically, a **taxonomy** is a way to group things together. Like a controlled vocabulary focuses on concepts, i n practice, a good taxonomy should be simple, easy to remember, and easy to use.*

For a best performance at UI we suggest - when possible - a taxonomy where each parent has 200 children or less.

Each folder in a taxonomy represents a "concept" and lists a couple of nodes (folders, documents, mails or records) associated within.

Keywords

A keyword, in the context of document management is a particular word or phrase that describes the contents. Setting keywords is the most simple and basic system to categorizing the nodes in your repository but at same time is the less powerfull. For large repositories we encourage using metadata groups.

Limitations of keywords usage:

- *When you have thousands of keywords searching by keywords can be inefficient (not able to refine search and you get a lot of results with same keywords, the users not followed a restricted way to catalog). A tag cloud diagram in this cases is usually an unusable graphical representation of keywords.*
- *It's difficult for the users setting keywords because they do not know all the keyword dictionary of the company.*
- *Users write keywords in similar way but not always the same (for example users use the words "building",*

"construction" and "edifice" indistinctly).

You can get best results when:

- *You limit the number of users who are able to set keywords.*
- *You use a controlled keywords dictionary (more information at http://www.w3.org/2004/02/skos/).*
- *You share a complete company keywords dictionary to your users. That helps understanding how is mapped the knowledge in your organization and helps users finding it.*

Categories

In OpenKM, the categories are a virtual taxonomy that helps on navigation of taxonomy nodes across other concepts. Each category node represents a "concept" and list a couple of nodes (folders, documents, mails or records) associated within.

We suggest using categories rather than keywords.

Metadata

Metadata groups are data that describes other data providing information about a certain nodes content. Metadata groups helps on creating controlled classification and a powerful way to locate nodes and refine query searchs.

With metadata groups you can simulate the keywords and categories features. We encourage expend some time investigating metadata feature and we suggest use it rather keywords and categories.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Plan your data organization

Before and intensive use of your application you should plan your data organization in the repository.

> *Take the information that follows only as a general idea about the basic steps you should consider before an intensive use of your application.*

A good analysis of your organization and the understanding of the application can help you finding the best way to solve your problems. OpenKM is not a closed application, something more than a couple of features focused on document management. OpenKM is a flexible application and can be used on several ways. It's good idea to expend some time understanding your organization, your business logic and process involved and how the OpenKM features can work together in your day-by-day helping your organization making it more efficient and saving money.

Some questions you should do yourself about your organization:

- *Where is your information and documents ?*
- *What kind of documents do you have ?*
- *Are your documents on electronic format or not ?*
- *What format of document do you have ? PDF, MS Office, OpenOffice, etc.*
- *How are you actually finding the document ? How would you like to find it, which terms you consider relevant ?*
- *Who can access the information and on which way ?*
- *Do you need a work flow process ?*

Which are the objective of the previous questions ?

- *Identify on a table all the distinct types of information you have (invoices, budgets, legal papers, etc.).*
- *Identify the format of the documents (PDF, DOC, etc.)*
- *Identify non digitalized information that should be processed (scanned). The digitalization process can be a basic process.*
- *Identify for each type of information (document type) what information is relevant (metadata). For example invoices usually have invoice number, range of date, customer among others. The metadata will help users to retrieve fast and efficiently the information.*
- *Identify the security that must be applied for each document type. Who can read, write or delete information. Also your security needs will help you on setting the level of the application log you need. OpenKM provide you a full audit on any user action that can be configured in a more or less restrictive way.*
- *Identify workflows and people involved in workflows.*

> *The goal is to create a plan with calendar and milestones.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Which are the goals you are looking for the implantation of the application ? Usually could be:

- *Set some order in my organization. Application "force" the users to work in some way, that benefits your company because helps you "setting rules" that all users must follow.*
- *Consolidate the intellectual capital. Consolidating all your company knowledge on a single point you have all the company knowledge available to be share among all your users and setting rules on it.*
- *Help on accomplish with legal regulation.*

Try to set and specification on a single document answering this questions and others. It will help you to plan how to organize the information into.

Good practice

- *Consider OpenKM as a project that affects all your departments and usually your customers.*
- *Choose the leaders.*
- *Do gradual implantation of the project. Can start with some department or group of users or for some business logic process. Think the implantation as a virus in your company. A good leader should demonstrate the benefits of the new and show to the other users. Users usually receipt any news on the company as something negative if they perceive the application as something is helping on their work (efficient search, saving time and solving real problems, etc.) will be the first interested on using it. One of the major goals of OpenKM is the intuitive UI what helps on this process (after two weeks the user feels comfortable using the application).*
- *Think easy, the best solution is usually finding the easiest way to solve problems.*
- *Listen your users and try to get them involved in the project.*
- *Periodically review your plan, try to refine and identify possible problems.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with the menu bar

At the menu bar on the top all available features are organized by type. The menu options is formed by all the pull down menus with their different options. If an option within a menu appears greyed-out, this option is not available

The menu bar options are:

- *File.*
- *Edit.*
- *Tools.*
- *Bookmarks.*
- *Templates.*
- *Help.*

The features described below are the default features. In your menu bar options can be shown others not described here if you have OpenKM extensions enabled.

Massive column in the tables, indicates you can check several nodes at the same time in browser panel to apply the same action.

File

Feature	Description	Massive
Find folder	*Finds folder in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*
Find document	*Finds a document in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*
Find mail	*Finds a mail in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*

Find record	*Finds a record in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*
Find similar document	*Finds similar documents in the repository. The feature is based on the search engine heuristic that is based on contents of the selected document.* *The search can be executed only on the context of the selected stack.* *This feature is only enabled when a document in Browser view is selected .*	*No.*
Create folder	*Creates a new folder into the selected node of the navigator panel.*	*No.*
Add document	*Uploads a new document into the selected node of the navigator panel.* *When the option "Import documents from ZIP" is checked you import the contents of a ZIP file respecting the hierarchy of the folder tree. If you have problems with filename encodings, please take a look at Client configuration. You can also upload RAR or JAR MIME-TYPE files.* *When the option "Notify to users" is checked you can send a mail notification to other application users.* *Optionally, based on **Administration** > Profiles configuration or **Administration** > Automation a wizard is shown. Wizard can require to you for setting metadata groups, categories or keywords.*	*No.*
Add record	*Creates a new record into the selected node of the navigator panel.*	*No.*
Download document	*Downloads the selected document in browser panel.*	*No.*
Download document	*Download the selected document in browser panel in PDF format.* *The feature is enabled when is possible convert the document to PDF format.*	*No.*

as PDF		
Send link	*Sends a URL to the selected document in browser panel by mail .*	*Yes.*
Send document attachment	*Sends the selected document in browser panel as an attachment by mail.*	*Yes.*
Generate download link	*Generates a download token of the selected document in browser panel and send the URL.* *For more information take a look at Download tokens.*	*No.*
Forward e-mail	*Forwards the selected mail in a browser panel.*	*No.*
Write mail	*Directly writes a new mail from user interface.*	*No.*
Create from template	*Creates a new document based on an existing one at templates space.*	*No.*
Export to ZIP	*Downloads the selected nodes compress into ZIP file. If you have problems with filename encodings, please take a look at Client configuration.*	*Yes.*
Start workflow	*Start a workflow on the selected node in browser panel.* *You can start a workflow on any node type (document, folder, mail or record).* *Optionally some workflows require you to fill a form as part of starting.* *The available workflows are set in **Administration** > Profiles.*	*No.*

Refresh	*Refresh the user interface.*	*No.*
File uploader	*Bulk upload feature.* *This feature needs to have Java enabled in your browser. More information at Bulk Uploader.* *If you're using a recent version of Goggle Chrome, keep on mind that Java applets won't work. Please take a look at Browser configuration for more info.*	*No.*
Restore	*Restores the selected node from the trash bin.*	*No.*
Purge	*Purges the selected node from the trash bin.*	*No.*
Purge trash	*Cleans the user trash bin.*	*No.*
Exit	*Exits the application.* *Users should always exit the application with this menu option and not only closing the web browser. Anyway after 30 minutes the web session of the user in the server will be automatically terminated.*	*No.*

Example of allowed queries on find folder, document, mail or record.

- *Add an asterisk "*****" as a placeholder for any unknown or wildcard terms, for example, "***2015**", "**2015***", "**20*15**" or "***2015***".*
- *Add an asterisk "**?**" as a placeholder for any unknown or wildcard character, for example the query "**?015**" will return occurrences where first character doesn't matter.*
- *The query "**2015**" is a synonym of "**2015***". On this search view application always add "*****" for you at the end.*

Edit

Feature	*Description*	*Massive*
Lock	*Locks the selected document in a browser panel.*	*Yes.*
Unlock	*Unlocks the selected document in a browser panel.* *Only the user who locked the document or an user with administration credentials (force unlock) can unlock the node.*	*Yes.*
Edit	*Edits a document (check-out).*	*Yes.*
Update	*Uploads a new document version (check-in).* *When uploading a new version of a document, you can notify users about the document changes. The application sends the notification by mail.*	*No.*
Cancel edit	*Cancels a document edition.* *Only the user who edited the document or user with administration credentials (force cancel edit) can cancel edition of the node.*	*Yes.*
Delete	*Deletes selected node.* *When a node has been deleted, you can restore it from the trash bin. Each user has its own trash.*	*Yes.*
Copy	*Copies the selected node.*	*Yes.*
Move	*Moves the selected node.*	*Yes.*
Rename	*Renames the selected node.* *You can cancel renaming using the ESC key.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Add note	*Adds a note to the selected node.*	*Yes.*
Add category	*Adds a category to the selected node.* *Categories are a virtual taxonomy space that helps on the navigation of taxonomy nodes across other concepts*	*Yes.*
Add keyword	*Adds a keyword to the selected node.* *When thesaurus is enabled you can choose terms from there (SKOS).*	*Yes.*
Merge PDF	*Merges two or more PDF files. The* ***result of the process is a new PDF documen****t.* *Musts chooses two or more PDF files to be merged.* *You can change the process order of the PDF documents.* *You can set the name of the new PDF document result of merging the selected PDF files.*	*No.*
Add metadata group	*Adds a metadata group to the selected node.*	*Yes.*
Update metadata group	*Updates a metadata group fields values of the selected node.*	*Yes.*
Remove metadata group	*Deletes the metadata group of the selected node.*	*Yes.*
Add subscription	*Subscribes to the selected node.*	*No.*

	You can subscribe on any node type (document, folder, mail or record). In case you are subscribed to node of type folder or record, you'll be notified by mail with any change in that node and children hierarchy. The user will receive notification based on the events definition of its own profile (only will receive notifications on events enabled from its own profile). More information about configuring notification events on Profiles.	
Remove subscription	*Un subscribe to the selected node.*	*No.*

*The document edition is based on the "**check-out / check-in**" model.*

*"**Check-out**" and "**check-in**" are common terms in many document management systems, including OpenKM. Their purpose is to prevent conflicts in an environment where multiple people might want to edit the same piece of content - in OpenKM a document - at the same time.*

*The term "**check-out**" describes the process of getting a version of a document. By checking out a document, a user can prevent others from editing that content.*

*The term "**check-in**" describes the process of updating a document to replace the previous version. By checking in the item, the user can allow others to edit the document, without needing to worry about overriding changes that others have made.*

When a document is checked out to a user, that user can work on that document, and other users cannot. Other users must wait for the user to check in the document before they can edit. While the document is checked out, other users usually can see only the last version that was checked in; they can't see the changes that the current user has made on the document while it's been checked out.

Tools

Feature	*Description*	*Massive*
Languages	*Changes the user interface language.*	*No.*
Skin	*Changes the user interface skin.*	*No.*
Debug console	*Shows the frontend console.*	*No.*
Show administration	*Shows the administration tab on a separate browser tab.* *This option is only available for Administrators users.*	*No.*

User configuration	*Shows the user configuration. You can change your password and mail address.*	*No.*
OCR data capture	*Applies Object Character Recognition on the selected node.* *The data captured is stored in a metadata group.* *This option is only available if you have enabled a OCR template. For more information about it take a look at **Administration** > OCR templates.*	*No.*
OMR	*Applies Object Mark Recognition on the selected node.* *The data captured is stored in a metadata group.* *This option is only available if you have enabled some OMR template. For more information about it take a look at **Administration** > OMR templates .*	*No.*
Web page import	*Captures a web page and import as PDF file.* *This option is only available if you have configured the parameter "**system.html2pdf**". More information at Preview parameters.*	*No.*
Convert	*Transforms the selected document to other format.* *Available options:* • *When **Save** checkbox is checked a new document is created in the repository.* • *When **Download** checkbox is checked the converted document is automatically download.*	*Yes.*
DXF preview	*When enabled the autocad extension feature, preview autocad files as DXF.*	*No.*

	*This option is only available if you have enabled the **Autocad preview** feature.*	
PDF preview	*When enabled the autocad extension feature, preview autocad files as PDF.* *This option is only available if you have enabled the **Autocad preview** feature.*	*No.*
Add new document encrypted	*Upload a new document and crypt.* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*
Encrypt document	*Encrypts the selected document into the repository.* *When a document is encrypted all previous version history will be removed (because previous versions history are not encrypted).* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*
Edit encrypted document	*Edits an encrypted document.* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*
Update encrypted document	*Uploads a new version of an encrypted document.* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*
Download decrypted document	*Downloads version decrypted of the selected encrypted document.* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Decrypt document	*Decrypt the selected document in the repository.* *This option is only available if you have enabled the **Cryptography** feature.*	*No.*
Send new message	*Sends new internal message between OpenKM and internal messaging.* *This option is only available if you have enabled the **Messaging** feature.*	*No.*
Sign document	*Applies digital signature to the selected document.* *This option is only available if you have enabled the **Digital signature** feature.*	*No.*
Export to Dropbox	*Exports the selected node to Dropbox.* *This option is only available if you have enabled the **Dropbox** feature.*	*Yes.*
Import from Dropbox	*Import data from Dropbox to selected folder at Navigator.* *This option is only available if you have enabled the **Dropbox** feature.*	*No.*
Export to Google Drive	*Exports the selected node to Goggle Drive.* *This option is only available if you have enabled the **Goggle Drive** feature.*	*Yes.*
Import from Google Drive	*Imports data from Goggle Drive to selected folder at Navigator.* *This option is only available if you have enabled the **Goggle Drive** feature.*	*Yes.*
Create new document	*Creates a new document and starts editing with Goggle Drive integration view.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	Goggle Drive integration allows collaborative edition with several users on edition at same time. *This option is only available if you have enabled the **Goggle Drive** feature.*	
Edit	*Edits the selected document with Goggle Drive integration view.* *Goggle Drive integration allows collaborative edition with several users on edition at same time.* *This option is only available if you have enabled the **Goggle Drive** feature.*	*No.*
HTML Editor	*You can enable or disable integrated HTML editor.* *This option is only available if you have enabled the **HTML Editor 4** feature.*	*No.*
Live Editor	*You can enable or disable the Live Edit feature. When enabled you edit the selected document in their native application (for example MS Office, OppenOffice or LibreOffice).* *This option is only available if you have enabled the **Live Edit** feature.* *This feature needs to have Java enabled in your browser. More information at Live Edit.* *If you're using a recent version of Goggle Chrome, keep on mind that Java applets won't work. Please take a look at Browser configuration for more info.*	*No.*
Manage contacts	*You can add or edit existing contacts.* *This option is only available if you have enabled the **Contact** feature.*	
Zoho edit	*You can editor document with integrated Zoho Editor.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The allowed documents are:

- *Ms Office (doc, xls)*
- *OpenOffice and LibreOffice (odt, ods)*
- *Text files.*
- *CSV files.*
- *HTML files.*
- *RTF files.*

*This option is only available if you have enabled the **Zoho** feature.*

Bookmark

Feature	*Description*	*Massive*
Home	*Go to home node.*	*No.*
Set default home	*Change default home for selected node.* *When user log in, the application opens in desktop view the home node of the user.* *By default home node is "**/okm:root**".* *You can select nodes in navigation and browser panels from desktop.* *More information at Working with desktop.*	*No.*
Add Bookmark	*Bookmark the selected node.* *You can select nodes in navigation and browser panels from desktop.* *More information at Working with desktop.*	*No.*
Edit Bookmark	*Show bookmark manager*	*No.*

From bookmark view you can see the list of nodes added in bookmark by user.

When you click on a bookmark node the application opens the node in desktop view.

Templates

*Template menu dynamically build a menu hierarchy based on "**/okm:templates**" nodes. When you click on any menu option at templates, automatically starts the process of creating a new node (usually a new document based on template document).*

Help

Feature	*Description*
Documentation	*Opens a new browser tab with official openkm documentation URL.*
Support request	*Opens a new browser tab with official community support URL.*
Public forum	*Opens a new browser tab with official public forum URL.*
Version notes	*Opens a new browser tab with application changelog URL*
OpenKM	*Opens a new browser tab with official OpenKM website URL.*
About OpenKM	*Shows a pop-up with information of application.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with the toolbar

The toolbar section has the most common features like download documents, create new folders, edit or upload files among others. The toolbar is made up of buttons with the main functions that the user will most frequently use. As with the Menu bar, occasionally the buttons will appear greyed-out, meaning that they are not available.

The features described below are the default ones. Your toolbar can shows others not described here if you have OpenKM extensions enabled.

Massive column in the tables, indicates you can check several nodes at the same time in a browser panel to apply the same action.

Feature	*Description*	*Massive*
Find folder	*Find folder in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*
Find document	*Finds a document in the repository.* *The search is executed only on the context of the selected stack.*	*No.*
Find mail	*Finds mail in the repository.* *The search is executed only on the context of the selected stack.*	*No.*
Find record	*Finds record in the repository.* *The search can be executed only on the context of the selected stack.*	*No.*
Find	*Finds similar document in the repository. The feature is based on search engine heuristic that*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

similar document	*based on contents of the selected document return similar documents.* *The search can be executed only on the context of the selected stack.* *This feature is only enabled when a document is selected in a Browser view.*	
Create folder	*Creates a new folder into the selected node of the navigator panel.*	*No.*
Download document	*Downloads the selected document in a browser panel.*	*No.*
Download document as PDF	*Downloads the selected document in a browser panel in PDF format.* *The feature is enabled when it is possible to convert the document to PDF format.*	*No.*
Lock	*Locks the selected document in a browser panel.*	*Yes.*
Unlock	*Unlocks the selected document in a browser panel.*	*Yes.*
Add document	*Uploads a new document into the selected node of the navigator panel.* *When the option "Import documents from ZIP" is checked you import the contents of a ZIP file respecting the hierarchy of the folder tree. If you have problems with filename encodings, please take a look at Client configuration.* *When the option "Notify to users" is checked you can send a mail notification to other application users.* *Optionally, based on **Administration** > Profiles configuration or **Administration** > Automation a wizard is shown. Wizard can require to you for setting metadata groups, categories or keywords.*	*No.*
Print	*Prints the selected document in a browser panel.*	*No.*

Edit	*Edits a document (check-out).*	*Yes.*
Update	*Uploads a new document version (check-in).*	*No.*
Delete	*Deletes selected node.*	*Yes.*
Cancel edit	*Cancels document edition.*	*Yes.*
Add metadata group	*Adds metadata group to the selected node.*	*Yes*
Delete metadata group	*Deletes the metadata group of the selected node.*	*Yes.*
Add subscription	*Subscribes to the selected node.* *You can subscribe to any node type (document, folder, mail or record). In case you are subscribed to a particular node of type folder or record, you'll be notified by mail when any change in that node and children hierarchy occurs. Keep on mind that in order to receive these mail notifications you have to configure which events you want to be notified in the user profile. See Profile Notification tab for more info.*	*No.*
Remove subscription	*Unsubscribes to the selected node.*	*No.*
Start workflow	*Starts a workflow on the selected node in a browser panel.* *You can start a workflow on any node type (document, folder, mail or record).*	*No.*

Refresh	*Refreshes user interface.*	*No.*
Home	*Goes to home node.*	*No.*
Scan	*Scanning.*	*No.*
Bulk upload	*Bulk upload feature.* *This feature needs to have Java enabled in your browser.* *More information at Bulk Uploader.*	*No.*
By default	*Restores default panels distribution.*	*No.*
Close tree	*Closes navigator panel.*	*No.*
Close properties	*Closes properties panel.*	*No.*
Expand tree	*Maximizes navigator panel.*	*No.*
Expand explorer	*Maximizes browser panel.*	*No.*
Expand properties	*Maximizes properties panel.*	*No.*
Previous node	*Goes to the last node visited.*	*No.*
History	*Manages history of nodes visited.*	*No.*

Next node	*Goes to the next node visited.*	*No.*
Stamp	*Applies a stamp on a document.* *This option is only available if you have enabled the **Stamp** feature.*	*No.*
Macro	*Executes a macro.* *This option is only available if you have enabled the **Macros** feature.*	*No.*
Menu options	*Displays an extended menu with custom options.* *This option is only available if you have enabled the **Menu options** feature.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with desktop

Almost all user's activity is done there. Users can work directly on folders, documents, mails or records.

The desktop is distributed in tree sections:

- ***Navigation** panel at left hand side.*
- ***Browser** panel at top center.*
- ***Properties** panel at bottom center.*

Navigation

The navigation at left hand side is an stack panel. Each stack contains a tree (folder and records hierarchy) structure from root that enables easily access documents and mails.

The navigation panel is divided on several stacks:

- *Taxonomy.*
- *Categories.*
- *Metadata.*
- *Thesaurus.*
- *Templates.*
- *My documents.*
- *E-mail.*
- *Trash.*

Each stack contains a tree (folder and records hierarchy) to help on organizing the nodes into the repository and help the user to locate fast and efficiently the data.

*The **taxonomy** stack is the space for collaborative work, that means is the space where several users modify and share the company data.*

*The **categories** stack is a virtual taxonomy space that helps on navigation of taxonomy nodes across other concepts.*

> *Categories is a hierarchical tree structure of category and subcategory folders. The idea is these folders and sub folders represents conceptual terms in a tree structure.*

*The **metadata** stack is a virtual taxonomy space that helps on the navigation of taxonomy nodes across the metadata field values.*

> *Metadata view is a hierarchical tree structure of metadata groups. Each possible metadata value is represented*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

as a tree node structure.

Only fields of type Select, SuggestBox and Checkbox are displayed (free text fields like Inputs or Text area are not shown).

You can create your own Stack Metadata Adapter visualization, more information at: Creating your own Stack Metadata Adapter.

*The **thesaurus** stack is a virtual taxonomy space that helps on the navigation of taxonomy nodes across SKOS concepts.*

*A **thesaurus** is a book that lists **words grouped together according to a similarity of meaning** (containing synonyms and sometimes antonyms), in contrast to a dictionary, which contains definitions and pronunciations.*

SKOS concepts are linked to many, but are represented as a tree structure.

*The **templates** stack is the space to set templates for all the company. That means is the space where company templates are set to be used by several users for creating new company documents.*

*The **My documents** stack is a private space for each user, that means that only a user is able to get access in this section and is the default where personal data is stored.*

*The **E-mail** stack is a private space for each user, that means only a user is able to access their data in this section and is the default place where are stored the mails of the user.*

*The **Trash** stack is a private space for each user, that means that only a user is able to get access in this section. When a user deletes a node it goes automatically to its own trash. When a user empty its own trash the document is definitively removed from the application.*

Administrators have no restriction for accessing all repository areas, including private user sections like "My Document", "E-mail" or "Trash".

The selected node from navigator panel determines the content shown at browser panel. All the nodes shown at the browser panel have the same parent (selected node on tree).

In case of virtual stacks (categories, metadata and thesaurus) the nodes shown at browser panel do not have the same parent (selected node on tree), really they are shown there because they have a relation (category, metadata or thesaurus value) with the selected node on tree.

*When you do a **right click** with the mouse button on a node at the tree is shown a **contextual menu**. The available menu options in contextual menu depending on selected stack.*

A complete description of all menu options are available at: Working with the menu bar.

*Available menu options at **Taxonomy** stack are:*

- *Create folder.*
- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Add document.*
- *Add record.*
- *Add bookmark.*
- *Set default home.*
- *Export to ZIP.*

*Available menu options at **Categories** stack are:*

- *Create folder.*
- *Delete.*
- *Move.*

*Available menu options at **Templates** stack are:*

- *Create folder.*
- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Export to ZIP.*

*Available menu options at **My document** stack are:*

- *Create folder.*
- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Export to ZIP.*

*Available menu options at **E-mail** stack are:*

- *Create folder.*
- *Delete.*
- *Rename.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Move.*
- *Copy*

*Available menu options at **Trash** stack are:*

- *Restore.*
- *Purge.*
- *Purge trash.*

Metadata and Thesaurus stack view has no menu options.

Those folders to which the user is subscribed will be displayed in a different color to differentiate them:

- *You are subscribed to a folder **without subfolders** into.*
- *You are subscribed to a folder **with subfolders** into.*
- *You are subscribed to a **read only** folder **without subfolders** into.*
- *You are subscribed to a **read only** folder **with subfolders** into.*

Browser

The browser panel at the top center shows a table of nodes (documents, folders, mails and records) that are in the selected node of the navigation panel.

*At the top of the browser panel there's a **path** that indicates the actual location in the navigation panel (for example "**/okm:root/**"). Below, there is a list of the nodes related with the selected node in the navigator panel.*

Usually are listed the nodes that have the selected node as parent, except on Categories, Metadata and Thesaurus view where the nodes listed are related with the selected node and have not the same parent.

By default the browser table shows the columns:

- *Status (icons display node status, for example if the node is locked is shown the **Lock** **icon**).*
- *Checkbox (to apply actions on several nodes at same time).*
- *Icon (indicate node type).*
- *Name (name of the node).*
- *Title.*
- *Language (only for documents).*
- *Size.*
- *Update date.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Author.*
- *Version (only for documents).*

Also in browser table can be shown extra columns based on metadata field values. For more information about how enable it take a look at: ***Administration*** *>* *Profiles.*

When you do a ***right click*** *on the mouse button at some node it is shown a* ***contextual menu****. You will have distinct menu options depending on the selected node type and the visible stack panel combination.*

A complete description of all menu options is available at: *Working with the menu bar.*

Available menu options when ***Taxonomy, My documents*** *stack is visible are:*

- *Download.*
- *Edit.*
- *Update.*
- *Cancel edit.*
- *Lock.*
- *Unlock.*
- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Convert.*
- *Add note.*
- *Add category.*
- *Add keyword.*
- *Add metadata group.*
- *Update metadata group.*
- *Merge PDF.*
- *Add bookmark.*
- *Set default home.*
- *Export to ZIP.*
- *OCR data capture.*
- *OMR data capture.*
- *Send link.*
- *Send document attachment.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Forward e-mail.*

*Available menu options when **Categories, Metadata** and **Thesaurus** stack is visiable are:*

- *Download.*
- *Edit.*
- *Update.*
- *Cancel edit.*
- *Lock.*
- *Unlock.*
- *Delete.*
- *Rename.*
- *Add note.*
- *Add category.*
- *Add keyword.*
- *Add metadata group.*
- *Update metadata group.*
- *Go to (document, folder, mail, record). Open node location in taxonomy view.*
- *Send link.*
- *Send document attachment.*
- *Forward e-mail.*

*Available menu options when **Templates** stack is visible are:*

- *Download.*
- *Edit.*
- *Update.*
- *Cancel edit.*
- *Lock.*
- *Unlock.*
- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Convert.*
- *Add note.*
- *Add category.*
- *Add keyword.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Add metadata group.*
- *Update metadata group.*
- *Merge PDF.*
- *Export to ZIP.*
- *Create from template.*
- *Add bookmark.*
- *Send link.*
- *Send document attachment.*
- *Forward e-mail.*

Create from template *helps you to create new documents based on existing templates. More information at: Creating templates.*

Available menu options when ***E-mail*** *stack is visible are:*

- *Delete.*
- *Rename.*
- *Move.*
- *Copy.*
- *Convert.*
- *Add note.*
- *Add category.*
- *Add keyword.*
- *Add metadata group.*
- *Update metadata group.*
- *Merge PDF.*
- *OCR data capture.*
- *OMR data capture.*
- *Send link.*
- *Send document attachment.*
- *Forward e-mail.*

Available menu options when ***Trash*** *stack is visible are:*

- *Restore.*
- *Purge.*

The ***checkbox*** *helps you to* ***select several nodes*** *at same time and* ***apply an action for all****.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

You can do actions on several nodes at the same time *(you have a checkbox in each row for selecting the node). The massive actions allowed are:*

- *Delete.*
- *Move.*
- *Copy.*
- *Add notes.*
- *Add categories.*
- *Add keywords.*
- *Add metadata groups.*
- *Change metadata group field values.*

On ***making a subscription*** *to node an* ***icon*** *will appear at the side of node indicating that* ***you are subscribed***. *Keep on mind that in order to receive these mail notifications you have to configure which events you want to be notified in the user profile. See Profile Notification tab for more info.*

When there is no checkbox selected the action will be applied to the selected node (blue row mark the selected node in browser view).

The Massive icon displays a contextual menu for doing quick actions:

Feature	*Description*
Select all	*Selects all the nodes.*
Select all folders	*Selects all the nodes of type folder.*
Select all document	*Selects all the nodes of type document.*
Select all mails	*Selects all the nodes of type mail.*
Select all records	*Selects all the nodes of type record.*
Remove selection	*Unselect all the nodes.*

Pagination

When ***pagination feature*** *is enabled is shown and extra toolbar:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Feature	*Description*
Show / hide folders	*Shows / hides nodes of type folder in the browser panel.*
Show / hide documents	*Shows / hides nodes of type document in the browser panel.*
Show / hide mails	*Shows / hides nodes of type mail in the browser panel.*
Show / hide records	*Shows / hides nodes of type record in the browser panel.*
Filter	*Filters nodes.*
Paginate *checkbox*	*Enables or disables the pagination feature.*
Limit *select*	*Sets the limit of nodes to be shown at the same time.*
Go to the beginning	*Goes to the beginning of the list.*
Go to previous	*Goes to previous list.*
Go to next	*Goes to next list.*
Go to the end	*Goes to the end of the list.*
Order *select*	*Sets the column value for that ordering.*
Reverse *checkbox*	*Reverses the ordering.*

Mouse features

- ***Double-click*** *on a node of type folder to* ***open the folder***.
- ***Double-click*** *on a node of type document to* ***download the document***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Right click*** *on a node will show a* ***contextual menu***.
- ***Drag & drop***. *You can drag & drop nodes keeping the left mouse button held down and moving to destination node in the Navigator panel.*

Properties

The properties panel at bottom center, show a collection of tabs that contains the grouped properties or features like preview associated with the selected node. The selected node can be a navigator node or a browser node.

Basically the selected node at the navigator panel determines the content shown at the browser panel and depending if the last selected node by the user is at navigator or browser panel the properties panel will shown the properties of the active node at navigator or browser.

The features described below are the default features. In your properties tab can be shown others not described here if you have OpenKM extensions enabled.

Tabs on nodes of type document

Tab	*Description*
Properties	*Displays properties of the selected document.* *From this view you can:* • ***Set the keywords*** *of the document.* • ***Change the title*** *of the document.* • ***Set relations between the document and categories***. *From this tab is shown a* ***list with the users subscribed to the node***. *For a quick node sharing, there are two special icons named* ***URL*** *and* ***WebDAV*** *when clicking on them, it will copy in your clipboard a browser or WebDAV URL.*
Notes	*Displays a list of notes related with the document. From this view you can add, edit or remove your own notes.* *Administrators can manage notes of other users.*
History	*Displays a list with the versions of the document.* *From this view you can:*

- ***Restore to a previous version*** *of the document.*
- ***Preview any document version*** *on the right panel.*
- ***Compare two versions of the document*** *on the preview panel.*
- ***Compact the history***.

The action of compacting the history will remove all the document version except the active. This action freezes repository size but must be take in consideration that this action can not be reverted.

Security

Displays the security grants of the document. From this view you can change the security grants.

- *Click on* ***Update button*** *to change the security grants.*

When "Recursive permission changes" checkbox is checked the changes will be applied recursivelly.

You should also be interested on reading:

- *Security configuration.*
- *Security configuration parameters.*
- *"**Advanced filter**" option on Profile General tab.*

Preview

Displays a pre visualization of the document.

Relations

Displays the relations of the document to other nodes (documents, folders, mails or records). From this view you can manage the relations of the document, adding or removing relation.

To set a new relation:

- *First must* ***select the type*** *from the relation type list.*
- *Then* ***choose one*** *of the defined relations.*
- ***Select a node in the repository***, *four icons for searching are shown: document, folder, mail or record.*

There're three types or relations:

- *Bidirectional*
- *Parent-child*

- *Many-to-many*

More information at Relation types.

Contacts	*Displays contacts associated with the document.* *From this view you can:* • ***Associate existing contacts*** *with the document.* • ***Add a new contact and associate it*** *with the document.* • ***Delete the association between*** *the contact and the document.* • ***Edit a contact*** *associated with the document.* • ***View contact details***. *This option is only available if you have enabled the **Contacts** feature.*
Activity log	*This view performs a complete audit trail of the document. From this view you can filter by specific actions.* *When the checkbox named **View** is not checked, the most common actions like **GET_CHILDREN_DOCUMENTS** are automatically filtered.* *The reason why most common actions are filtered when checkbox name View is not checked is that these events usually create a large list of actions and often are not relevant from the audit trail point of view.* *This option is only available if you have enabled the **Activity log** feature.*
Forum	*Displays a forum associated with the document.* *From this view you can:* • ***Add a new topic***. • ***Post a reply*** *of the topic.* • ***Quote a reply*** *of the topic.* • ***Edit a reply*** *of the topic.* • ***Delete a topic***. *To delete or edit a post of the topic you must be the owner or have administrator grants.*

	*This option is only available if you have enabled the **Forum** feature.*
Wiki	*Display a wiki associated with the document.* *From this view you can:* • ***Create a wiki**.* • ***Update a wiki**.* • ***Show the history** of wiki versions.* • ***Restore a version** of the wiki.* *This option is only available if you have enabled the **Wiki** feature.*
Workflow	*Display the workflows associated with the document.* *From this view you can:* • *Show the **detail** of the workflow variables.* • ***Show a graph** where it is marked the actual node.* *This option is only available if you have enabled the **Workflow** feature.*
Text extracted	*Displays the text extracted of the document.* *Only when the document has been processed by text extractor worker you will see the text. For more information: **Administration** > Statistics.* *This option is only available if you have enabled the **Text extracted** feature.*

Tabs on nodes of type folder

Tab	*Description*
Properties	*Displays properties of the selected folder.* *From this view you can:* • ***Change the folder style**.*

- ***Set the keywords*** *of the folder.*
- ***Change the title*** *of the folder.*
- ***Set relations between the folder and categories*****.**

This tab shows a ***list with the users subscribed to the node*****.**

For a quick node sharing, there are two special icons named ***URL*** *and* ***WebDAV*** *when clicking on them, it will copy in your clipboard a browser or WebDAV URL.*

More information: Folder style.

Notes

Displays a list of notes related with the folder. From this view you can add, edit or remove your own notes.

Administrators can manage other user's notes.

Security

Displays the security grants of the folder. From this view you can change the security grants.

- *Click on* ***Update button*** *to change the security grants.*

When the "Recursive permission changes" checkbox is checked the changes will be applied recursivelly.

You could be interested on reading:

- *Security configuration.*
- *Security configuration parameters.*
- *"****Advanced filter****" option on Profile General tab.*

Relations

Displays the relations of the folder to other nodes (documents, folders, mails or records). From this view you can manage the relations of the folder, adding or removing relation.

To set a new relation:

- *First must* ***select the type*** *from the relation type list.*
- *Then* ***choose one*** *of the defined relations.*
- ***Select a node in the repository****, four icons for searching are shown:* *document,* *folder,* *mail or* *record.*

	There're three types or relations: • *Bidirectional* • *Parent-child* • *Many-to-many* *More information at: Relation types.*
Contacts	*Displays contacts associated with the folder.* *From this view you can:* • ***Associate existing contacts** with the folder.* • ***Add new contacts and associates** with the folder.* • ***Delete association between** the contact and the folder.* • ***Edit a contact** associated with the folder.* • ***View contact details**.* *This option is only available if you have enabled the **Goggle contacts** feature.*
Activity log	*This view performs a complete audit trail of the folder. From this view you can filter by specific actions.* *When the checkbox named **View** is not checked, the most common actions like **GET_CHILDREN_DOCUMENTS** are automatically filtered.* *The reason why most common actions are filtered when checkbox name View is not checked is that these events usually create a large list of actions and often are not much relevant from the audit trail point of view.* *This option is only available if you have enabled the **Activity log** feature.*
Forum	*Display a forum associated with the folder.* *From this view you can:* • ***Add new topic**.* • ***Post a reply** to the topic.* • ***Quote a reply** tof the topic.*

	• ***Edit a reply*** *of the topic.* • ***Delete a topic***. *To delete or edit a post of the topic you must be the owner or have administrator grants.* *This option is only available if you have enabled the* ***Forum*** *feature.*
Wiki	*Displays a wiki associated with the folder.* *From this view you can:* • ***Create a wiki***. • ***Update a wiki***. • ***Show the history*** *of wiki versions.* • ***Restore a version*** *of the wiki.* *This option is only available if you have enabled the* ***Wiki*** *feature.*
Workflow	*Display the workflows associated with the folder.* *From this view you can:* • *Show the* ***detail*** *of the workflow variables.* • ***Show a graph*** *where is marked the actual node.* *This option is only available if you have enabled the* ***Workflow*** *feature.*

Tabs on nodes of type mail

Tab	*Description*
Properties	*Display properties of the selected mail.* *From this view you can:* • ***Set the keywords*** *of the mail.* • ***Change the title*** *of the mail.* • ***Set relations between the mail and categories***.

*This tab shows a **list with the users subscribed to the node**.*

*For quick node sharing, there are two special icons named **URL** and **WebDAV** when clicking on them, it will copy in your clipboard a browser or WebDAV URL.*

Notes

Displays a list of notes related with the mail. From this view you can add, edit or remove your own notes.

Administrators can manage notes of other users.

Security

Displays the security grants of the mail. From this view you can change the security grants.

- *Click on **Update button** to change the security grants.*

When "Recursive permission changes" checkbox is checked the changes will be applied recursivelly.

You might be interested on reading:

- *Security configuration.*
- *Security configuration parameters.*
- *"**Advanced filter**" option on Profile General tab.*

Preview

Displays a pre visualization of the mail.

From this view you can also download the mail attachments or copy them to a destination folder.

Relations

Displays the relations of the mail to other nodes (documents, folders, mails or records). From this view you can manage the relations of the document, adding or removing relation.

To set a new relation:

- *First must **select the type** from the relation type list.*
- *Then **choose one** of the defined relations.*
- ***Select a node in the repository**, for it are shown four icons for searching a document, folder, mail or record.*

There're three types or relations:

- *Bidirectional*
- *Parent-child*
- *Many-to-many*

More information at: Relation types.

Contacts	*Displays contacts associated with the mail.* *From this view you can:* • ***Associate existing contacts*** *with the mail.* • ***Add a new contact and associate it*** *with the mail.* • ***Delete the association between*** *the contact and the mail.* • ***Edit a contact*** *associated with the mail.* • ***View contact details***. *This option is only available if you have enabled the* ***Goggle contacts*** *feature.*
Activity log	*This view performs a complete audit trail of the mail. From this view you can filter by specific actions. When the checkbox named* ***View*** *is not checked, the most common actions like* ***GET_CHILDREN_DOCUMENTS*** *are automatically filtered.* *The reason why most common actions are filtered when checkbox name View is not checked is that these events usually create a large list of actions and often they are not relevant from the audit trail point of view.* *This option is only available if you have enabled the* ***Activity log*** *feature.*
Forum	*Displays a forum associated with the mail.* *From this view you can:* • ***Add a new topic***. • ***Post a reply*** *to the topic.* • ***Quote a reply*** *of the topic.* • ***Edit a reply*** *of the topic.* • ***Delete a topic***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	To delete or edit a post of the topic you must be the owner or have administrator grants. *This option is only available if you have enabled the **Forum** feature.*
Wiki	*Displays a wiki associated with the mail.* *From this view you can:* • ***Create a wiki**.* • ***Update a wiki**.* • ***Shows the history** of wiki versions.* • ***Restore a version** of the wiki.* *This option is only available if you have enabled the **Wiki** feature.*
Workflow	*Display the workflows associated with the mail.* *From this view you can:* • *Show the **detail** of the workflow variables.* • ***Show a graph** where is marked the actual node.* *This option is only available if you have enabled the **Workflow** feature.*

Tabs on nodes of type record

Tab	*Description*
Properties	*Displays properties of the selected record.* *From this view you can:* • ***Set the keywords** of the record.* • ***Change the title** of the record.* • ***Set relations between the mail and categories**.* *This tab is shown as a **list with the users subscribed to the node**.* *For a quick node sharing, there are two special icons named **URL** and **WebDAV** when clicking on them will copy in your clipboard a browser or WebDAV URL.*

Notes	*Displays a list of notes related with the record. From this view you can add, edit or remove your own notes.* *Administrators can manage notes of other users.*
Security	*Displays the security grants of the record. From this view you can change the security grants.* • *Click on **Update button** to change the security grants.* *When "Recursive permission changes" checkbox is checked the changes will be applied recursivelly.* *You might be interested on reading:* • *Security configuration.* • *Security configuration parameters.* • *"**Advanced filter**" option on Profile General tab.*
Relations	*Displays the relations of the record to other nodes (documents, folders, mails or records). From this view you can manage the relations of the record, adding or removing relation.* *To set a new relation:* • *First must **select the type** from the relation type list.* • *Then **choose one** of the defined relations.* • ***Select a node in the repository**, four icons for searching are shown document, folder, mail or record.* *There're three types or relations:* • *Bidirectional* • *Parent-child* • *Many-to-many* *More information at: Relation types.*
Contacts	*Displays contacts associated with the record.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

From this view you can:

- ***Associate existing contacts*** *with the record.*
- ***Add a new contact and associate it*** *with the record.*
- ***Delete the association between*** *the contact and the record.*
- ***Edit a contact*** *associated with the record.*
- ***View contact details***.

This option is only available if you have enabled the ***Goggle contacts*** *feature.*

Activity log

This view performs a complete audit trail of the record. From this view you can filter by specific actions.

When the checkbox named ***View*** *is not checked, the most common actions like* ***GET_CHILDREN_DOCUMENTS*** *are automatically filtered.*

The reason why most common actions are filtered when checkbox name View is not checked is that these events usually create a large list of actions and often they are not relevant from the audit trail point of view.

This option is only available if you have enabled the ***Activity log*** *feature.*

Forum

Displays a forum associated with the record.

From this view you can:

- ***Add a new topic***.
- ***Post a reply*** *to the topic.*
- ***Quote a reply*** *of the topic.*
- ***Edit a reply*** *to the topic.*
- ***Delete a topic***.

To delete or edit a post of the topic you must be the owner or have administrator grants.

This option is only available if you have enabled the ***Forum*** *feature.*

Wiki

Display a wiki associated with the record.

From this view you can:

	• ***Create a wiki***. • ***Update a wiki***. • ***Show the history*** *of wiki versions.* • ***Restore a version*** *of the wiki.* *This option is only available if you have enabled the* ***Wiki*** *feature.*
Workflow	*Display the workflows associated with the record.* *From this view you can:* • *Show the* ***detail*** *of the workflow variables.* • ***Show a graph*** *where is marked the actual node.* *This option is only available if you have enabled the* ***Workflow*** *feature.*

Shortcuts

Shortcuts can be used from desktop view.

Available shortcuts:

Shortcut	*Description*
F2	*Rename a node.*
CTRL+SHIFT+C	*Copy a node.*
CTRL+SHIFT+X	*Cut a node.*
CTRL+SHIFT+V	*Paste a node.*
CTRL+D	*Downloading.*
CTRL+G	*Show UI testing popup.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CTRL+Z	*Show UI log.*
CTRL+N	*Create folder.*
F5	*Refresh desktop view.*
INSERT	*Add new document.*
ALT+B	*Bookmark a node.*
F1	*Show help.*
CTRL+ALT+Q	*Exit application.*
ALT+DELETE	*Delete a node*

Shortcuts are configurable from Profiles.

Working with search

The search tab enables the query the repository. The search view is distributed in four sections:

- *Searches panel.*
- *Search panel.*
- *Control panel.*
- *Results.*

Searches panel

The Search panel at left hand side is an stack panel.

> *Once a search has been done, it can be saved for future sessions by entering a name that identifies it and clicking on the "Save search" button.*

This panel have two stacks:

- *Stored searches.*
- *User news.*

***Stored searches** shows a list of user's searches that has been saved to be use later. With a double-click on stored search, you can execute the search again. With right click on a stored search will be shown a contextual menu with this options:*

Feature	*Description*
Run	*Run stored search.*
Delete	*Delete an stored search.*
Share query	*Share an stored search.* *This option is only available if you have enabled the **Messaging** feature.*

> *The stored searches and users news are only visible for the user who created them.*

***User news** shows a list of user searches that are used as push service (usually executed every 30 minutes). These searches are periodically executed and the results are shown at **Dashboards > News** panel.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

User news helps users to be periodically updated about repository news based on his own queries.

Will be shown only new results that have appeared since the last time the query was executed.

Search panel

This panel at the top center gives access to different searches to query the content of the application:

- ***Default search** is a full-text search. Default search is also available from the top right corner on the screen.*
- ***Advanced search** enables you to search on keywords, metadata and additional criteria such as location in the application.*

*You can **switch between default or advanced** search **checking or un checking** the "**View advanced mode**" checkbox.*

*Default search engine configuration uses **stemming** and **stopwords**.*

- ***Stemming** is the term used in linguistic morphology and information retrieval to describe the process for reducing inflected (or sometimes derived) words to their word sterm, base or root form generally a written word form. The stem needs not to be identical to the morphological root of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. For instance, if you type "fishing" in a full-text search field, search results will return documents with "fishing", "fish", "fisher". In that case, the stem is "fish".*
- ***Stopwords** usually refer to the most common words in a language, there is no single universal list of stop words used by all processing of natural language tools, and indeed not all tools even use such a list. The stopword are removed from search. For example "the", "is", "at", "which" and "on" are stopwords.*

Operators and wildcards

*By default internally the **search engine use the "AND" operator**.*

You can use wildcards:

- *Add an asterisk "*****" as a placeholder for any unknown or wildcard terms, for example, "***2015**", "**2015***", "**20*15**" or "***2015***".*
- *Add an asterisk "**?**" as a placeholder for any unknown or wildcard character, for example the query "**?015**" will return occurrences where first character doesn't matter.*
- *The query "**2015**" is a synonim of "**2015***". On this search view application always add "*****" for you at the end.*

Default search

Default search is a full-text search.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Default search shows results nodes only in "/okm:root" context, results from other context are automatically filtered.

The search takes in consideration:

- *Content (document content).*
- *Title.*
- *Keywords.*
- *Name.*
- *Subject (mail subject).*
- *Body (mail content)*
- *Notes.*
- *UUID (node UUID).*

Always at the top right corner you have quick access with Quick search box feature.

Advanced search

Advanced search enables you to search on keywords, metadata and additional criteria such as location in the application.

The advanced search organizes the query fields on tree tabs:

- *Basic (most common fields).*
- *Advanced.*
- *Metadada.*

The available criteria from basic view :

Field	*Description*	*Wildcard allowed*
Context	*Sets the context where is executed the query.* *Available context:* • *Taxonomy.* • *Templates.* • *My documents.* • *E-mail.* • *Trash.*	*No.*

Content	*The content of the node.* *Document content or mail body.*	*Yes.*
Name	*The name of the node.*	*Yes.*
Title	*The title of the node.*	*Yes.*
Keyword	*The associated keywords of the node.*	*Yes.*
Language	*The language of the document.* *Only the nodes of typed document has language value.*	*Yes.*
User	*The creator.*	*No.*
Date range	*The range of dates where the node has been modified (or created).* *Each selected date is set to 00 hours, 00 minutes, 00 seconds and 00 miliseconds. That means if you want to do a query of a node modified the day 2015-08-15 must set a range from "2015-08-15" to "2015-08-16" (internally you're querying from "2015-08-15 00:00:00" to "2015-08-16 00:00:00").* *It is not mandatory to set both dates for range ("**from**" or the "**to**"), you can set only one date.*	*No.*

The available criteria from advanced view :

Field	*Description*	*Wildcard allowed*
Folder	*Restrict results to some folder.* *When Recursive checkbox is checked also considers the subfolders.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Category	*Restricts results to some category. The nodes must be associated to a specific category.*	*No.*
Type	*Restrict the types of node:* • *Document* • *Folder* • *Mail* • *Record* *When Mail checkbox is checked are shown extra fields (from, to , subject)*	*No.*
MIME type	*Set the document MIME type.*	*No.*
Notes	*The content of the notes.*	*Yes.*
From	*The from mail field.* *This field is only available if you have **checked the Mail checkbox**.*	*Yes.*
To	*The to mail field.* *This field is only available if you have **checked the Mail checkbox**.*	*Yes.*
Subject	*The subject mail field.* *This field is only available if you have **checked the Mail checkbox**.*	*Yes.*

From metadata view you can filter by any metadata field from existing metadata groups:

- *Click on* ***Add metadata group*** *button.*
- ***Choose one metadata group*** *from the list.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Choose a property*** *from the list.*
- *Click* ***Add*** *button.*

> *On almost input and text area fields you can use wildcards.*

Control panel

Control panel at the right top corner has several features:

Featureright	*Description*
Switch to Advanced mode checkbox	*When checked is shown the advanced mode search view.*
Results compact view checkbox	*When checked the results are shown in a compact table with minimal details.*
View metadata groups checkbox	*When checked are shown metadata groups associated to each node in results view.* *This option is only available when "Results compact view" is not checked.*
Save as user news checkbox	*When checked if the user saves a search, will be set to user news search.*
Save search button	*Once a search has been done, it can be saved for future sessions by entering a name that identifies it and clicking on the "Save search" button.* *Depending if "Save as user news" checkbox is checked or not will be saved as "stored search" or "user news".*
Page results lists	*Set the limit of nodes to be shown in the results list.*
Clean button	*Resets the search.*
Search	*Executes the search.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Results	*Indicates the number of results (total).*
Go to previous	*Goes to previous list.*
Go to next	*Goes to next list.*
Export to CSV	*Downloads a CSV file with results.*
Export to ZIP	*Downloads a ZIP file with results.*

Results

The Results panel at the bottom center shows a table of nodes (documents, folders, mails and records) what are the result of querying the repository. Results panel are shown in:

- *Compact view (a table with a single row for each node).*
- *Full view (a detailed description for each node).*

Stars indicates the relevance of the document found. This value is established by the Search engine according to the search field and the results found. The list of documents found is ordered from most to least relevant.

When you do a right click on compact view will be shown a contextual with the options:

Feature	*Description*
Download	*Downloads the node.* *Only shown in document or mail nodes.*
Go to document	*Goes to the document location.* *Only shown in document nodes.*
Find similar documents	*Find similar document in the repository. The feature is based on search engine heuristic that based on contents of the selected document return similar documents.*

	Only shown in document nodes. *The search is executed only on the context of the selected stack.* *This feature is only enabled when is selected a document in Browser view.*
Go to folder	*Goes to folder location.* *Only shown in folder nodes.*
Go to e-mail	*Goes to mail location.* *Only shown in mail nodes.*
Go to record	*Goes to the record location.* *Only shown in record nodes.*
Preview	*Previews the node.* *Only shown in document or mail nodes.*

From full view you see detailed information of each node and also are available some features:

Feature	*Description*
Find similar documents	*Finds similar documents in the repository. The feature is based on search engine heuristic that based on contents of the selected document return similar documents.* *Only shown in document nodes.* *The search be executed only on the context of the selected stack.* *This feature is only enabled when is selected a document in Browser view.*
Download	*Download the node.*

	Only shown in document or mail nodes.
Notes	*Show notes. From this view can manage the notes of the node.*
Preview	*Preview the node.* *Only shown in document or mail nodes.*
Options	*Show extra mail extra fields (cc, bcc, reply, extra fields).* *Only shown in document or mail nodes.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with Lucene queries

With this new feature you will be able to create complex Lucene queries, so you won't be limited by the current search form fields. But keep on mind this can be difficult unles you know how to write these queries. Let's define a couple of items:

- ***Terms**: A query is broken up into terms and operators. A term is a single word like "cat" or "dog". Multiple terms can be combined with boolean operators for a more specific query. For example:*

  ```
  cat AND dog
  ```

- ***Fields**: When you perform a Lucene seach, you can specify a field or use the default field. In OpenKM the default field contains the document text extracted. You can search by a field writing the field name, followed by a colon ":" and a term. For example, this query will search for all nodes in taxonomy which name is "animals":*

  ```
  context:okm_root AND name:animals
  ```

> *You can search for a phrase using quotes:*
>
> ```
> name:"big animals"
> ```

Term modifiers

Lucene support different term modifiers to create complex searches.

Wildcard searches

You can use single and multiple character wildcard searches within terms.

- ***Single**: To perform a single character wirldcard search use the "?" symbol. For example, to search for "text" or "test":*

  ```
  te?t
  ```

- ***Multiple**: To perform a multiple characted wildcard, use the "*" symbol. For example, to search for "test", "tests" or "tester":*

  ```
  test*
  ```

> *You can't use a "*" or "?" at the begining of a query.*

Proximity searches

Lucene also supports finding words that are in a specific distance. To perform a proximity search use the tild "~" symbol and the end of the phrase.

For example, to search for a "cat" and "dog" within 10 words of each other in a document:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
"cat dog"~
```

Range searches

Range queries allow to match documents whose field values are between an specified lower and upper bound. Sorting is done lexicographically.

For example, to look for documents which field "name" is between "cat" and "horse" but not will include these terms:

```
name:{cat TO horse}
```

If you want these two terms to be included, use this query:

```
name:[cat TO horse]
```

Inclusive range queries are denoted by square brackets. Exclusive range queries are denoted by curly brackets.

Boolean operators

You can combine several terms using boolean operators. Lucene supports "AND", "+", "OR", "NOT" and "-".

Boolean operators must be in UPPERCASE.

Working with Lucene queries

With this new feature you will be able to create complex Lucene queries, so you won't be limited by the current search form fields. But keep on mind this can be difficult unles you know how to write these queries. Let's define a couple of items:

- ***Terms**: A query is broken up into terms and operators. A term is a single word like "cat" or "dog". Multiple terms can be combined with boolean operators for a more specific query. For example:*

  ```
  cat AND dog
  ```

- ***Fields**: When you perform a Lucene seach, you can specify a field or use the default field. In OpenKM the default field contains the document text extracted. You can search by a field writing the field name, followed by a colon ":" and a term. For example, this query will search for all nodes in taxonomy which name is "animals":*

  ```
  context:okm_root AND name:animals
  ```

You can search for a phrase using quotes:

```
name:"big animals"
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Term modifiers

Lucene support different term modifiers to create complex searches.

Wildcard searches

You can use single and multiple character wildcard searches within terms.

- ***Single**: To perform a single character wirldcard search use the "?" symbol. For example, to search for "text" or "test":*

```
te?t
```

- ***Multiple**: To perform a multiple characted wildcard, use the "*" symbol. For example, to search for "test", "tests" or "tester":*

```
test*
```

You can't use a "" or "?" at the begining of a query.*

Proximity searches

Lucene also supports finding words that are in a specific distance. To perform a proximity search use the tild "~" symbol and the end of the phrase.

For example, to search for a "cat" and "dog" within 10 words of each other in a document:

```
"cat dog"~
```

Range searches

Range queries allow to match documents whose field values are between an specified lower and upper bound. Sorting is done lexicographically.

For example, to look for documents which field "name" is between "cat" and "horse" but not will include these terms:

```
name:{cat TO horse}
```

If you want these two terms to be included, use this query:

```
name:[cat TO horse]
```

Inclusive range queries are denoted by square brackets. Exclusive range queries are denoted by curly brackets.

Boolean operators

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

You can combine several terms using boolean operators. Lucene supports "AND", "+", "OR", "NOT" and "-".

Boolean operators must be in UPPERCASE.

The "OR" operator is the default conjuntion operator: this means that if there is no boolean operator between two terms, the "OR" operator is used. These two queries are equivalents:

```
cat animal
```

And this one:

```
cat OR animal
```

AND

The AND operator matches documents where both terms exists. The symbol "&&" can be also used to replace the word "AND".

Let's look for documents with both "cat" and "animal" words in the content:

```
cat AND animal
```

+

This is called the required operator and force that the term placed after the "+" to be included in the results.

For example, to search for documents that must contain "animal" and may contain "cat":

```
+animal cat
```

NOT

The NOT operator excludes documents that contains the term after the "NOT". The symbol "!" can be used to replace the word "NOT".

For example, to search for documents that contains "animal" but not "cat":

```
animal NOT cat
```

The NOT operator cannot be used with just one term.

-

This operator excludes documents which contains the term given after the "-" character:

For example, to seach documents which contains "animal" but not "cat":

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
animal -cat
```

Grouping

Lucene supports using parentheses to group clauses to form sub queries. This can be very useful if you want to control the boolean logic for a query.

For example, to search for "cat" or "dog" and "animal" use this query:

```
(car OR dog) AND animal
```

Escaping special characters

Lucene supports escaping some special characters that are part of the query syntax using the slash "\" before the character to be escaped. This is the current list of special characters:

```
+ - && || ! ( ) { } [ ] ^ " ~ * ? : \
```

The "OR" operator is the default conjuntion operator: this means that if there is no boolean operator between two terms, the "OR" operator is used. These two queries are equivalents:

```
cat animal
```

And this one:

```
cat OR animal
```

AND

The AND operator matches documents where both terms exists. The symbol "&&" can be also used to replace the word "AND".

Let's look for documents with both "cat" and "animal" words in the content:

```
cat AND animal
```

+

This is called the required operator and force that the term placed after the "+" to be included in the results.

For example, to search for documents that must contain "animal" and may contain "cat":

```
+animal cat
```

NOT

The NOT operator excludes documents that contains the term after the "NOT". The symbol "!" can be used to replace the word

"NOT".

For example, to search for documents that contains "animal" but not "cat":

```
animal NOT cat
```

The NOT operator cannot be used with just one term.

-

This operator excludes documents which contains the term given after the "-" character:

For example, to seach documents which contains "animal" but not "cat":

```
animal -cat
```

Grouping

Lucene supports using parentheses to group clauses to form sub queries. This can be very useful if you want to control the boolean logic for a query.

For example, to search for "cat" or "dog" and "animal" use this query:

```
(car OR dog) AND animal
```

Escaping special characters

Lucene supports escaping some special characters that are part of the query syntax using the slash "\" before the character to be escaped. This is the current list of special characters:

```
+ - && || ! ( ) { } [ ] ^ " ~ * ? : \
```

Working with metadata fields

*Because OpenKM use lucene restricted characters like ":", the field name must be sanitized. For example, a field named in OpenKM as "**okp:consulting.text**", should be sanitized as "**okp.consulting.text**" replacing the character ":" by single ".".*

```
okp.consulting.text:value
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with the dashboard

The dashboard panel provides a set of dashboards, each user has a set of personal dashboard that displays the information relevant for him and helps to have a global view of the application activity.

The dashboard panel :

- ***User view**. Displays the actions that have been carried out in the system*
- ***General view**. Displays a global view of application activity.*
- ***News**. Displays information based on the scheduled user search queries.*
- ***Mail**. It's a virtual navigations across all the repository of mails.*
- ***Mail account**. Helps users to manage their own mail accounts.*
- ***Workflow**. Users work in company business process tasks.*
- ***Keyword map**. Displays a tag cloud based on all the repository keywords.*
- ***Tasks**. Helps on creating, administrate, assign and follow basic company task.*
- ***Calendar**. Helps on following the status of the basic company task.*
- ***Messaging**. Displays internal OpenKM messaging.*
- ***Forum**. Displays a forum.*
- ***Wiki**. Displays a wiki.*

The results listed in dashboard boxes depends on the security level access for each user.

User view

In the user's information view, users can access some interesting queries about the last user operations.

The information is organized in several boxes:

Box	*Description*
Locked documents	*Displays a list with the actual documents locked by the user.*
Locked records	*Displays a list with the actual records locked by the user.*
Edited documents	*Displays a list with the actual documents checkout by user.*
Last download documents	*Displays a list with the last 20 documents downloaded by the user.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Subscribed documents	*Lists subscriptions into documents by the user.*
Subscribed folders	*Lists subscriptions into folders by the user.*
Subscribed mails	*Lists subscriptions into mails by the user.*
Electronic mails	*Displays a list with the last 20 mails uploaded by the user.*
Attachments	*Displays a list with the last 20 documents attached into mails uploaded by user.*
Last modified documents	*Displays a list with the last 20 documents modified by the user.*
Last created documents	*Displays a list with the last 20 documents created by the user.*
Last created folders	*Displays a list with the last 20 folders created by the user.*
Last created records	*Displays a list with the last 20 records created by the user.*

Each box indicates the number of new items pending to be viewed by the user.

When a user clicks on a row, application will open up automatically the repository destination node.

General view

In the general information view, users can access some interesting queries about the last repository operations done by all users.

The information is organized in several boxes:

Box	*Description*
Last week top view documents	*Displays a list with the last 20 documents most viewed during last week.*
Last month top viewed documents	*Displays a list with the last 20 documents most viewed during last month.*
Last week top modified documents	*Displays a list with the last 20 documents most modified during last week.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Last month top modified documents	*Displays a list with the last 20 documents most modified during last month.*
Last created documents	*Displays a list with the last 20 documents created.*
Last modified documents	*Displays a list with the last 20 documents modified.*
Last created folders	*Displays a list with the last 20 folders created.*
Last created records	*Displays a list with the last 20 records created.*

Each box indicates the number of new items pending to be viewed by the user.

When a user clicks on a row, the application will open up automatically the repository destination node.

News

The News view display information based on scheduled user search queries. Each query is shown into a separate box. As a push service (usually executed every 30 minutes) the results are refreshed.

More information about registering "Users news" at: Working with search.

Mail

Mail view is a virtual navigations across all the repository of mails. The view is organized in sections:

- *Toolbar.*
- *Mail browser.*
- *Preview.*

Toolbar

Toolbar is at the top of the Mail panel.

Feature	*Description*

Show / hide mails without attachments	*Shows / hides nodes of type mail without attachments in a browser panel.*
Show / hide mails with attachments	*Shows / hides nodes of type mail with attachments in a browser panel.*
Filter	*Filter mails.* *Available filtering options:* • *Subject (allow wilcards).* • *From (allow wilcards).* • *Range of dates*
Limit *select*	*Sets the limit of mails to be shown at same time.*
Go to the beginning	*Goes to the beginning of the list.*
Go to previous	*Goes to previous list.*
Go to next	*Goes to next list.*
Go to the end	*Go to the end of the list.*
Order *select*	*Sets the column value for that ordering.*
Reverse *checkbox*	*Reverses the ordering.*

Mail browser

The Mail browser panel at center show a table of mails. To preview a mail, click on it.

Preview

The Preview panel at right side display mail preview.

The Preview panel is organized in three section:

- *Mail fields.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Message tab.*
- *Headers tab.*

With ***Options*** *icon at the top right in "Mail fields panel" you can show extra fields (cc, bcc, reply and extra fields).*

In the Message tab is shown the mail body contents.

In the Headers tab is shown the mail headers contents.

Mail accounts

In the Mail accounts view, are shown the mail user accounts. These mail accounts are used by the application to periodically import mail to te application repository.

> *The features and configuration are quite similar to manage users and mail account. The difference is that from the dashboard view the user only can manage their own mail accounts.*
>
> *More information at Manage user mail accounts.*

Workflow

In the Workflow panel are shown the workflows on execution. The Workflow panel is organized in two sections:

- *Workflow lists.*
- *Workflow detail.*

Workflow lists

In the Workflow lists panel are shown the workflows on execution organized in two boxes:

- *Pending tasks.*
- *Unassigned pending task.*

> *The workflows at "Pending task" box has yet to be assigned to the logged user and are waiting for him.*
>
> *The workflows at "Unassigned pending task" box has not yet beng assigned to a single user. There is a group of users who can auto assign the workflow themself (one of them is the logged user). The workflow is waiting for this auto asignation.*

Workflow detail

In the Workflow detail panel itis shown the workflow details. From this view you can see low level information data like "process instance", "process definition" or "data" and also ***go to the next workflow task completing the workflow form*** *.*

To complete a workflow task:

- ***Click on workflow at Pending task*** *box. It will be shown the workflow form.*
- ***Fill in the workflow form***.
- ***Click on any of the available buttons*** *to submit the values.*

Keyword map

The Keyword map view shows a tag cloud based on all the repository keywords. The Keyword map is organized in four sections:

- *Toolbar.*
- *Browser.*
  - *Tag cloud.*
  - *Results.*
- *Keywords list.*
- *Results controller.*

Toolbar

The toolbar is on top.

Feature	*Description*
Low detail	*Displays result nodes in low detail.*
Medium detail	*Displays result nodes in medium detail.*
Full detail	*Displays result nodes in full detail.*
Keywords	*Shows keywords used for filtering.*
Keyword selector	*The suggestbox selector helps to quickly choose the keywords for filtering.*
Clean	*Reset Keyword map view.*

Browser

The Browser panel is at the center. The panel is divided in two sections, at the top the ***Tag cloud*** *(visual representation of text*

data) and at the bottom the ***Results*** *table within the nodes that comply with the filtering rules.*

Available features on results :

- ***Go to document.***
- ***Go to folder.***
- ***Go to e-mail.***
- ***Go to record.***

In the "Medium" and "Full" detail view each node display its own associated keywords in two colors blue (for what are used on filtering) and grey.

Clicking on blue keyword you will remove from filtering rules.

Clicking on grey keyword you will add to filtering rules.

Keywords lists

The Keywords lists is at the right side. The keywords list is an stack panel organized in tree stacks:

- *Related keywords.*
- *Top Keywords.*
- *All Keywords.*

In the "Related keywords" stack are listed the related keywords based on the actual filtering rules. In the "Top Keywords" stack are listed the most commonly used keywords and finally in "All keywords" are list all the keywords.

Results controller

The results controller is at te bottom.

Feature	*Description*
Context selected	*Choose the context.* *Available context are:* • *Taxonomy.* • *Templates.* • *My documents.* • *E-mail.* • *Trash.* • *All repository.*

Page results	*Set the limit of the nodes to be shown at the same time.*
Results	*Displays the number of results.*
Go to previous	*Go to previous list.*
Go to next	*Go to next list.*

Tasks

The Tasks feature allows you to create, manage and assign tasks to users (or roles). The Tasks panel is organized in three sections:

- *Toolbar.*
- *Filter.*
- *Browser.*

Toolbar

Feature	*Description*
Task list	*Shows a task list view.*
States lists	*Manages the task status.* *Available features:* • *Add new state.* • *Edit.* • *Delete.* *States with "**Finish**" value active are used to indicate task end of process execution.*
Types list	*Manage task types.* *Available features:* • *Add new type.*

- *Edit.*
- *Delete.*

*You can disable a type, setting "**Active**" field to false.*

Project list	*Manage projects.*

Available features:

- *Add new project.*
- *Edit.*
- *Delete.*

*You can disable a project, setting "**Active**" field to false.*

Filter

The Filter on the left side helps users to filtering task by:

- *Assigned tasks.*
- *Active tasks.*
- *Closed tasks.*
- *State.*
- *Project.*
- *Type.*

*Also from this panel you can **Add new task states, projects or task types**.*

Browser

*The Browser at the right side list tasks based on filtering rules applied at the filter panel. From the top right corner you can **Add new task**. Also for each task you can **Edit** or **Delete** tasks. Tasks can be edited or deleted by the task owner (the user who created the task) or users with **ROLE_ADMIN** or **ROLE_TASK_MANAGER_ADMIN** roles.*

There's a tool tip feature which allows to show all task data from this view when the cursor is over the subject task.

Upon the creation or edition of a task a mail notification is sent to the users to inform them that a new task has been assigned.

Users with an assigned task which are not the owners - creator - only are allowed to modify the status, the task

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

progress or add link between the task and nodes.

Task fields:

Field	*Description*	*Mandatory*
Subject	*Sets the subject of the task.*	*Yes.*
Start	*Sets the starting date.*	*No.*
Ends	*Sets the ending date.*	*No.*
Repeat	*Checks whether the task will be periodically repeated or not.*	*No.*
Progress	*Sets the task progress.*	*No.*
Status	*Sets the status.*	*Yes.*
Project	*Sets the project.*	*Yes.*
Type	*Sets the type.*	*Yes.*
Users	*Users assigned to the task*	*Yes.*
Roles	*Roles are converted to users assigned to the task.*	*Yes.*
Find folder	*Links folders to the task.*	*No.*
Find document	*Links documents to the task.*	*No.*
Find mail	*Links mails to the task.*	*No.*
Find record	*Links records to the task.*	*No.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Add note	*Adds a note.*	*No.*

Calendar

Helps on following the status of the basic company task. There are three views available a Monthly, Weekly and Daily calendar.

There's a tool tip feature which allows to show all task data from this view when the cursor is over the subject task.

You can dynamically manage tasks, the available options are:

- *Drag & drop to move tasks in calendar.*
- *Double click on day (or hour when viewing calendar in daily view) to create a new task.*
- *Click on a task to display the data of task (from this view you can also* **Edit** *the task).*

Users with assigned task which are not the owners - creator - only are allowed to modify the status, the task progress or add link between the task and nodes.

Messaging

Messaging helps you proposing a nodes subscription to the users, sharing queries and sending text messages between users.

The Message panel is organized in three sections:

- *Message type*
- *Browser*
- *Preview*

Message type

Message type is on left side. Message type is an stack panel with the following options:

- *Propositions.*
- *Proposed queries.*
- *Messages received.*
- *Messages sent.*

Feature	Description
Propositions	*Displays users who have proposed subscriptions to nodes.*
Proposed queries	*Displays users who have shared their search queries.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Messages received	*Displays users who have sent messages to the user.*
Messages sent	*Displays users who you have sent messages.*

> *Bold users indicate you have pending information to be reviewed.*

Browser

Browser panel is on the top right. The panel list the messages based on "Message type" selection.

> *Bold message indicate you have still not previewed.*
>
> *Right clicking on a message will show a contextual menu with* ***Delete*** *action.*

Preview

Preview panel is on the bottom right. In this panel the message is previewed.

Feature	*Description*
Subscribe	*Subscribes to a node.* *Available when "Proposition" stack is visible.*
Execute search	*Executes shared search.* *Available when the "Proposed queries" stack is visible.*
Accept	*Accepts a shared search.* *Available when "Proposed queries" stack is visible.*

Forum

From the forum view you can ***manage all discussions*** *on nodes and also* ***create new forums*** *not linked with any node.*

Feature	*Description*	*Need administrator grants*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

New forum	*Adds a new forum.*	*Yes.*
Edit forum	*Edits a forum.*	*Yes.*
New topic	*Adds a new topic to an existing forum.*	*No.*
Post reply	*Posts a reply in an existing topic.*	*No.*
Quote	*Quotes a posting.*	*No.*
Edit post	*Edits a posting.*	*No when you are the owner.*
Delete	*Deletes a post.*	*No when you are the owner.*

Wiki

Display the main wiki page.

Field / Property	*Description*
Update	*Edits wiki contents.*
History	*Wiki page history.* *Available options:* • *View older page.* • *Restore previous version.*
Unlock	*Cancels wiki edition.*
Delete	*Deletes wiki page.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working in status bar

*The **Status** panel at the bottom provides general information. The status panel is organized in:*

- *OpenKM news alert.*
- *The connected user.*
- *User repository size.*
- *Chat.*
- *Alerts.*
- *HTML documents on edition.*
- *Message panel.*

OpenKM news alert

When there are OpenKM news is shown a blink ***alert icon** in the left side.*

> *Click on* ***OpenKM news icon** to display the news popup.*

The connected user

In left side is shown Username.

User repository size

In the user repository size is shown the total repository size of the user (documents and mails owned by the user).

Chat

You can ***connect** and* ***disconnect** user from internal application chat.*

> *By default Chat feature is disabled. More information about Chat feature at **Administration** > Profiles.*

Alerts

Click on any icon for opening related dashboard view.

Alert	*Description*
Locked nodes	*Displays the number of locked nodes by user.*
Document edited	*Displays the of edited documents by user.*

Subscription	*Displays the number of nodes subscribed by user.*
User news	*Displays the number of users news pending to be reviewed by user.*
Workflow tasks	*Displays the number of workflows tasks pending by user.*
Workflow pooled tasks	*Displays the number of workflows pooled tasks pending to be assigned.*
Proposed subscriptions	*Displays the number of proposed subscriptions by other users.* *This option is only available if you have enabled the **Messaging** feature.*
Shared queries	*Displays the number of shared queries pending to be approved by the user.* *This option is only available if you have enabled the **Messaging** feature.*
Internal messages	*Displays the number of internal messages sent by other users pending to be reviewed by the user.* *This option is only available if you have enabled the **Messaging** feature.*

HTML documents on edition

When a HTML document is edited from OpenKM there is a button for each document that helps on switching the document in editor.

You can edit several HTML documents at same time, and switching the document in editor helps on copy & paste between files.

*This option is only available if you have enabled the **HTML** feature.*

Message panel

In the right side are shown application text messages (for example while previewing a document are shown messages about the preview stages).

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

User email signature

There is an option to add a signature for emails created by OpenKM.

*Just place a HTML file called **signature.html** inside your personal folder "**/okm:personal/username**" and when you create a new email this content will be used as your signature.*

You can download an example here: signature.html

You can create a signature with any HTML valid code.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Bulk Uploader

The Bulk upload feature helps you to insert or update multiple documents at the same time without having to go to Add new document and select each document individually.

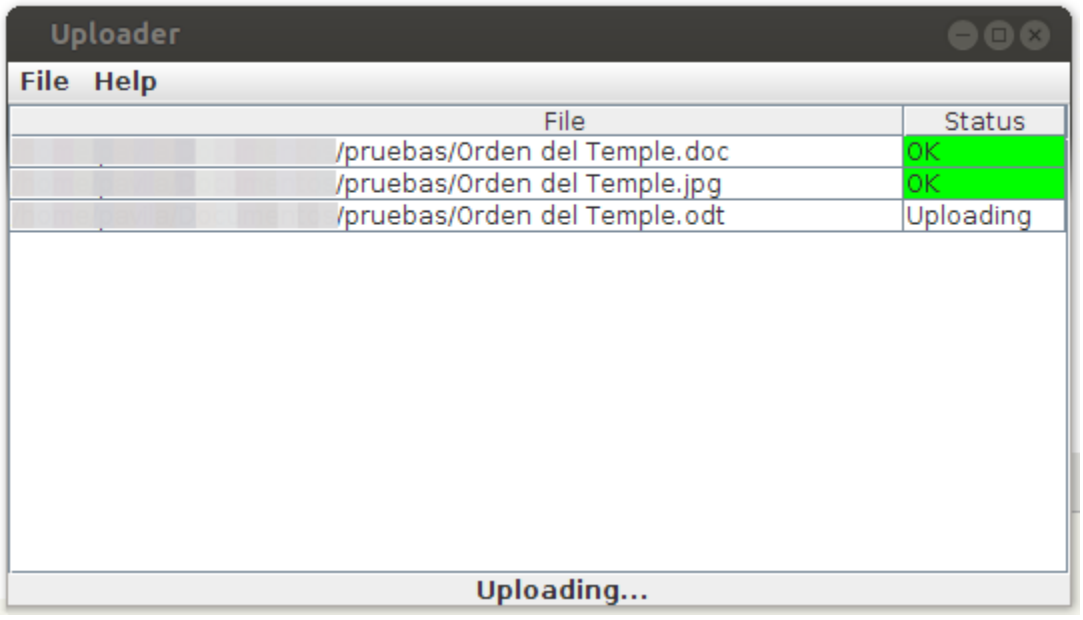
The feature is available from Menu bar (see Working with the menu bar) and Toolbar (see Working with the toolbar).

Uploader detects when the document to upload already exists in the repository, in that case, it will ask you if want to add this new content as a new document version.

Since OpenKM 6.4.29 this extension is not implemented as a Java Applet anymore. We had to migrate to Java Web Start due to incompatibilities between Java Applets and latest browser versions. So, when you want to upload several documents using this extension a JNLP file will be downladed and needs to be executed by Java. You have to agree about executing this app, and after that the Java application will be shown.

Usage

- ***Choose node*** *(folder or record) on your repository.*
- *Click on* ***Bulk upload icon****. Will be shown the Uploader Window.*



- ***Drag and drop files or folders*** *from your desktop computer* ***to the Uploader*** *window. The files and folders will be automatically uploaded to OpenKM, to your currently selected folder path*

*If a problem occurs while uploading a document, it will show an "**Error**" on the status column.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Live Edit

This extension make very easy to edit documents in OpenKM.

*This option is only available if you have enabled the **Live Edit** feature. More information at Enable extensions.*

Since OpenKM 6.4.29 this extension is not implemented as a Java Applet anymore. We had to migrate to Java Web Start due to incompatibilities between Java Applets and latest browser versions. So, when you want to edit a document a JNLP file will be downladed and needs to be executed by Java. You have to agree about executing this app, and after that the Java application will be shown.

Use

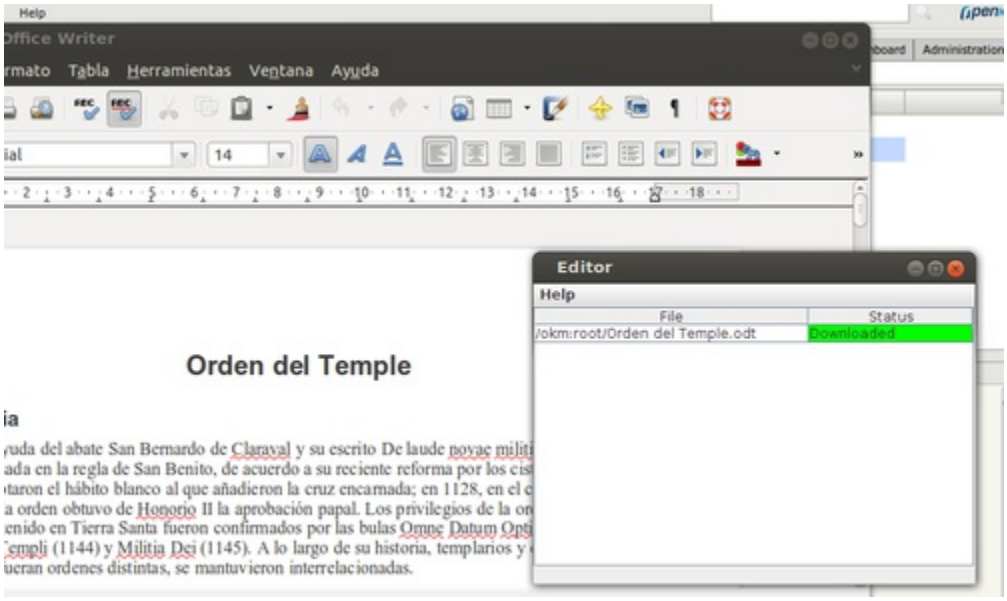
*When this extension is enabled, the action of the **Edit**, **Update** and **Cancel edit** buttons change and will be used to edit the document in this new way.*

- *Choose a document.*
- *Click on **Edit icon**. **Will be opened** an **Editor console** (do not close it) and your **default editor** in your computer.*

*Do not close the "**Editor**" console until you have finished your edition, otherwise your local changes will not be saved to OpenKM.*

*Once the **Edit** button is clicked, the document will be downloaded locally, opened with the default editor and marked as checkout. Of course, if you want to edit a .doc document you need to have installed Microsoft Office or OpenOffice / LibreOffice in your computer*

Warning message



Every time you save the document will be sent to OpenKM to preserve the changes. But the OpenKM document won't be modified until you click on the ***Update*** *toolbar button.*

- *Edit the document locally and save when finished, then close the application.*
- *Click on* ***Update icon.*** *Will be shown a confirmation dialog.*

*If you close the local application editor (MS Word, MS Excel, OpenOffice, LibreOffice, etc.) and want to continue modifying the document, "**double click**" in the desired document row at the "**Editor console**" window.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

HTML editor extension

The application comes with an integrated HTML editor.

> *This option is only available if you have enabled the **HTML Editor 4** feature.*
>
> *You can edit several HTML documents at the same time and switch the editor between them for copying and paste operations.*

Features

Feature	*Description*
Update	*Upload a new document version (check-in).* *When uploading a new version of a document, you can notify users about document changes. The application send the notification by mail.*
Cancel edit	*Cancel document edition.*
Link document	*Add link to a document.*
Link folder	*Add link to a folder.*
Add image	*Add image.*
Download link	*Add download link.*
Code	*Add code section.*

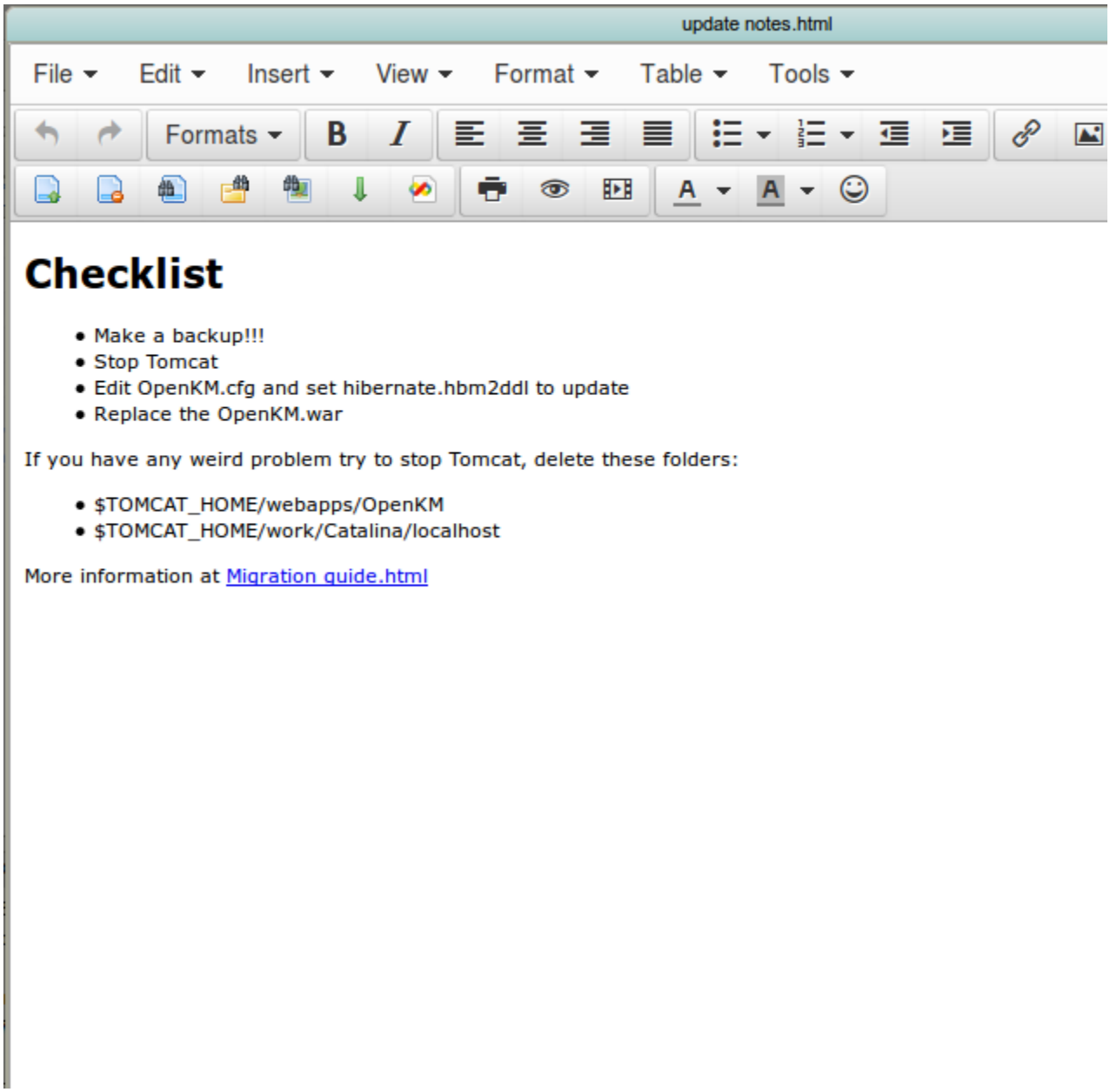
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

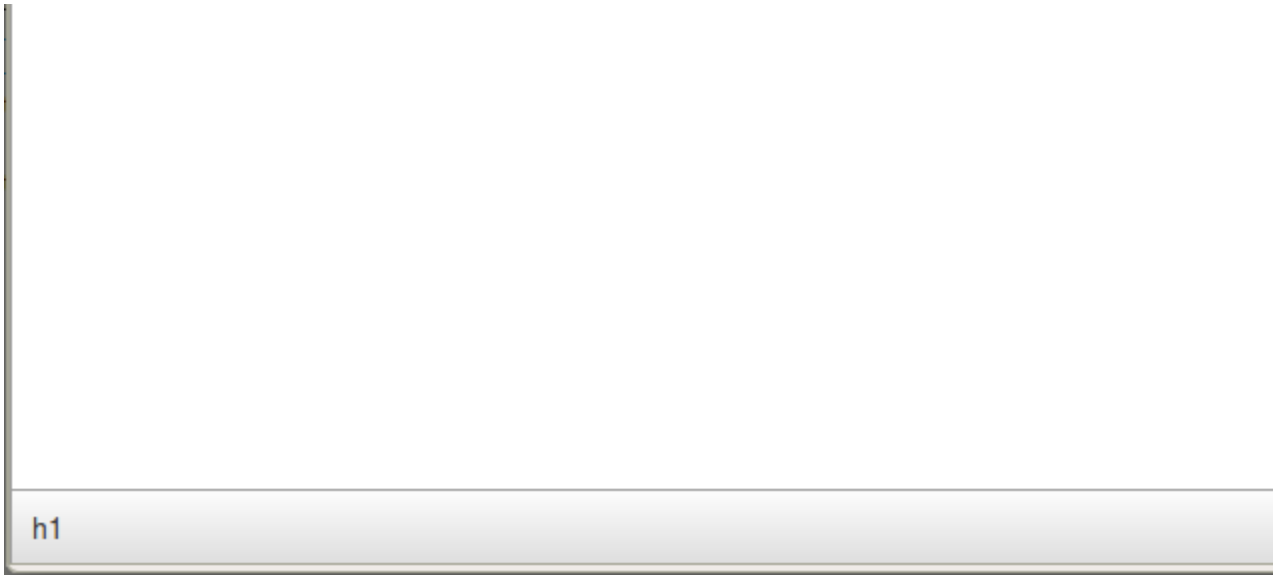
You can navigate across linked documents and folders or download documents from Preview view.

Edit HTML

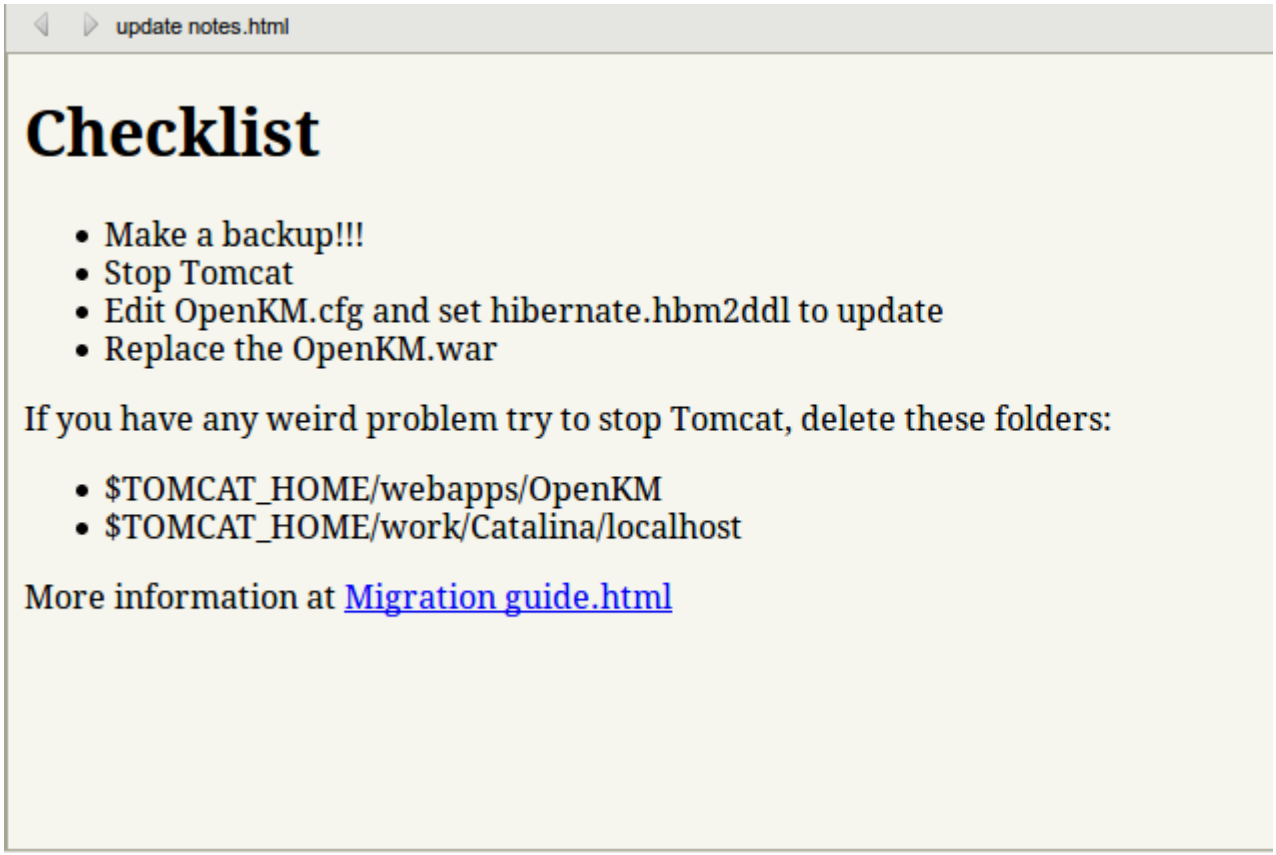
- ***Choose HTML document***.
- *Click on* ***Edit icon***.
- *Modify the content.*
- *Click on* ***Update icon***.

Editor panel





Preview panel



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutoCAD preview extension

AutoCAD previewer allows one to perform basic operations like zoom in/out, rotate, show/hide layers and print. AutoCAD preview extension allows to view AutoCAD files using SVG and JavaScript to display and manipulate graphics.

*This option is only available if you have enabled the **AutoCAD** feature.*

This feature needs third-party software installation. Ask your OpenKM technical staff for it.

Toolbar

Feature		*Description*
	Change icon Page	*Allows to change to previous or next icon page.*
	Layer List	*List of the layers of the drawing.*
	Print	*Print the document.*
	Zoom In	*Performs a zoom in.*
	Zoom Out	*Performs a zoom in.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

	Zoom Extends	*Allows you to move the zoom in zone.*
	Zoom Window	*Allows you to draw a window for do zoom in.*
	Toggle Black/White	*Allows you to toggle black/white color in the drawing.*
	Background Color	*Allows you to select a specific color for the background of the drawing.*
	File Load	*Not implemented yet.*
	Search Text	*Allows you to search individual text string in the drawing.*
	About	*Information about CADViewer JS version.*
	Layer Off	*Turn layer off.*
	Swap Layers	*Allows to swap between layers.*
	All Layers On	*Turn all layers on.*
	Measurement	*Allows to select to points in the drawing to measure the distance between them. Needs to calibrate measurement first.*
	Calibrate Measurement	*Allows to calibrate measurement to define units of measure in drawing.*
	Toggle Birds-Eye Window	*Open the birds-eye window.*
	Adjust Line Thickeness	*Adjust the line thickness of the drawing.*
	Save Redline	*Save current red lines.*

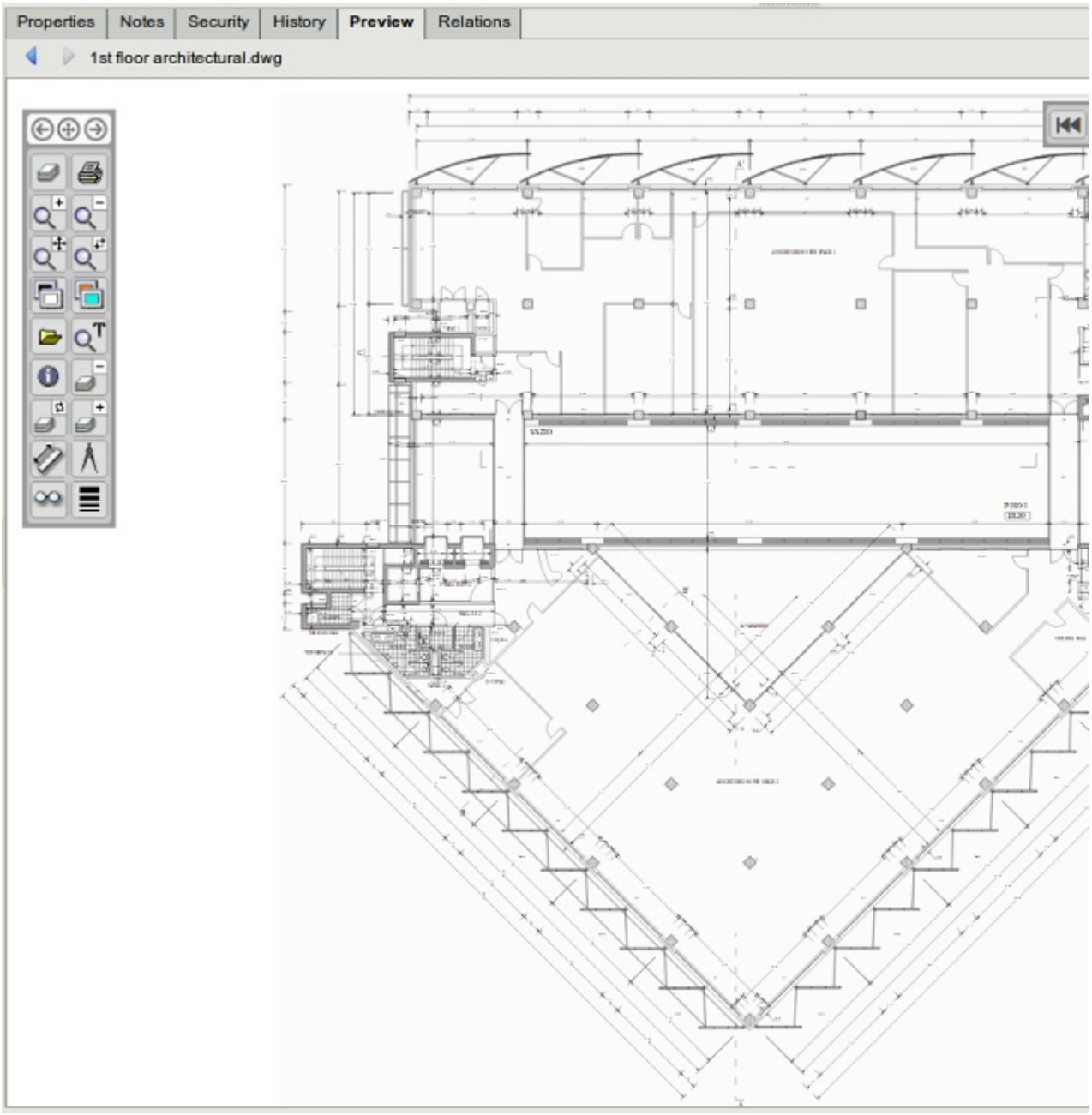
	Load Redline	*Allows to load red lines previously saved.*
	Delete Single Redline	*Allows to delete one red line by selecting it in the drawing.*
	Delete Last Redline	*Delete last red line.*
	Redline Color	*Allows to change the color of the red line.*
	Redline Thickness	*Allows to change the thickness of the red line.*
	Make Stickynote	*Allows to insert an sticky note.*
	Clear Visible Redlines	*Clear all visible red lines.*
	Freehand Redline	*Allows to insert a red line with free form.*
	Redline Cloud	*Allows to insert a red line with a cloud shape.*
	Redline Filled Polygon	*Allows to insert a red line with a filled polygon shape.*
	Redline Polyline	*Allows to insert a red line with polyline shape.*
	Redline Filled Rectangle	*Allows to insert a red line with a full rectangle shape.*
	Redline Rectangle	*Allows to insert a rectangle red line.*
	Redline Arrow	*Allows to insert a red line with an arrow.*
	Redline Text	*Allows to write a text and select where to place it in the drawing.*

	Zoom Extents	*Allows to move the zoom-in zone.*
	Zoom Window	*Allows to draw a window for do zoom-in.*

	Feature	*Description*
	First Page	*Go to the first page in the drawing.*
	Previous Page	*Go to the previous page in the drawing.*
	Next Page	*Go to the next page in the drawing.*
	Last Page	*Go to the last page in the drawing.*

Default preview

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Cryptography extension

Cryptography helps you to set additional protection to your documents.

*This option is only available if you have enabled the **Cryptography** feature.*

More information at Enable extensions.

This feature needs to have Java enabled in your browser. For security reasons the procedures of encrypting and decrypting are always done from the user computer side.

If you're using a recent version of Google Chrome, keep on mind that Java applets won't work. Please take a look at Browser configuration for more info.

When a document is encrypted you can not preview or search by contents.

Cryptography features are only available from Tools menu.

Feature	*Description*
Upload new document encrypted	*Upload a new document encrypted.*
Encrypt document	*Encrypt existing repository document.*
Edit encrypted document	*Edit existing encrypted repository document.*
Update encrypted document	*Upload a new version of an encrypted document.*
Download decrypted document	*Download version decrypted of the selected encrypted document.*
Decrypt document	*Decrypt the selected document into the repository.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Digital Signature extension

Digital Signature helps you to sign documents with installed certificates.

*This option is only available if you have enabled the **Digital Signature** feature.*

You can sign documents in repository or new documents.

Only PDF documents can be signed or which are able to be converted to PDF format.

More information at Enable extensions.

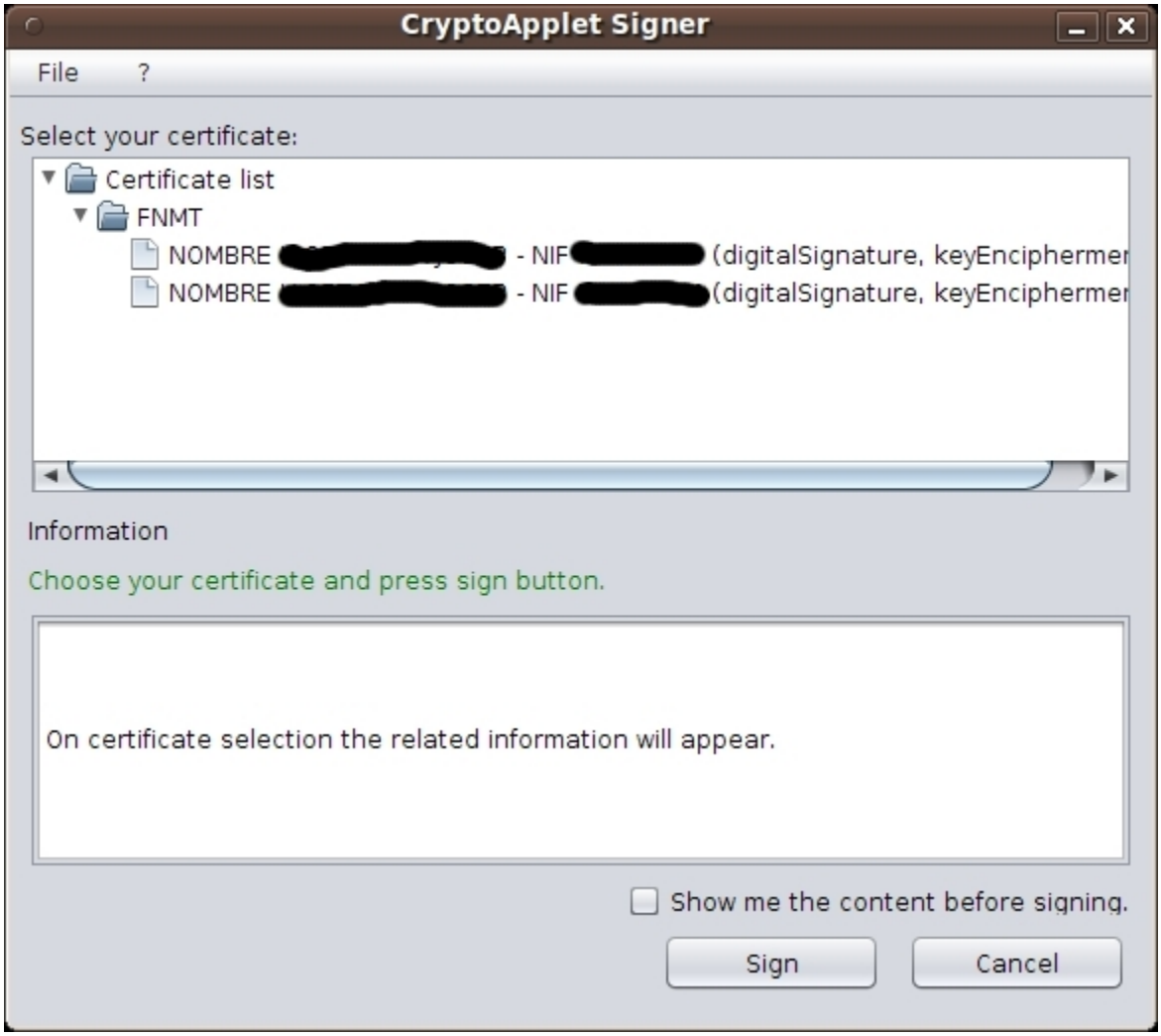
This feature needs to have Java enabled in your browser. For security reasons the procedures of encrypting and decrypting are always done from the user computer side.

If you're using a recent version of Google Chrome, keep in mind that Java applets won't work. Please take a look at Browser configuration for more info.

Sign document in repository

- *Choose the document.*
- *Click on* ***Sign document*** *menu option at tools menu. Will be shown a popup.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



- *Choose a certificate.*
- *Click on* ***Sign button****.*

> *As part of setting a Digital Signature a comment "signed" to the document. You can view it from history tab.*

Upload and sign new document

- *Click on* ***Add document*** *icon.*
- *Choose file from your computer.*
- ***Check*** *the* ***Sign Document checkbox****.*
- *Click on* ***Send button****.*

> *Digital Signature is not compatible with "Import document from ZIP" option.*

Digital Signature viewer extension

*Digital Signature viewer helps you to **view** and **verify** signed documents.*

> *This option is only available if you have enabled the **Digital Signature viewer** feature.*

Verify a signed document

- *Choose a signed document.*
- ***Show Digital Signature** viewer tab.*

Features:

Feature	*Description*
Integrity	*Display the result of checking the digital signature.*
Trust	*Display the result of checking if signer certificate is trusted.*
Revocation	*Display the result of checking revocation list from the certificate.*
Validity	*Display the result of checking certificate validation or signing time.*
Download the signed version	*You can download the signed document.*
Certificate information	• *You can download the certificate.* • *Display the subject.* • *Display the issuer.*
Validity information	*Display the signing time.*
Signed properties	• *Display signing reason.* • *Display signing location.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Document expiration extension

Document expiration is a notification system alert which can be used to advice users nearly document expiration and expired documents.

Add document expiration

- ***Choose a document***.
- ***Add*** *metadata group "**Expiration**".*
- ***Fill the form***.
- *Click* ***Add button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Extratab extension

The ExtraTab allows to easily add new tabs with a name and specific URL.

*This option is only available if you have enabled the **ExtraTab** feature.*

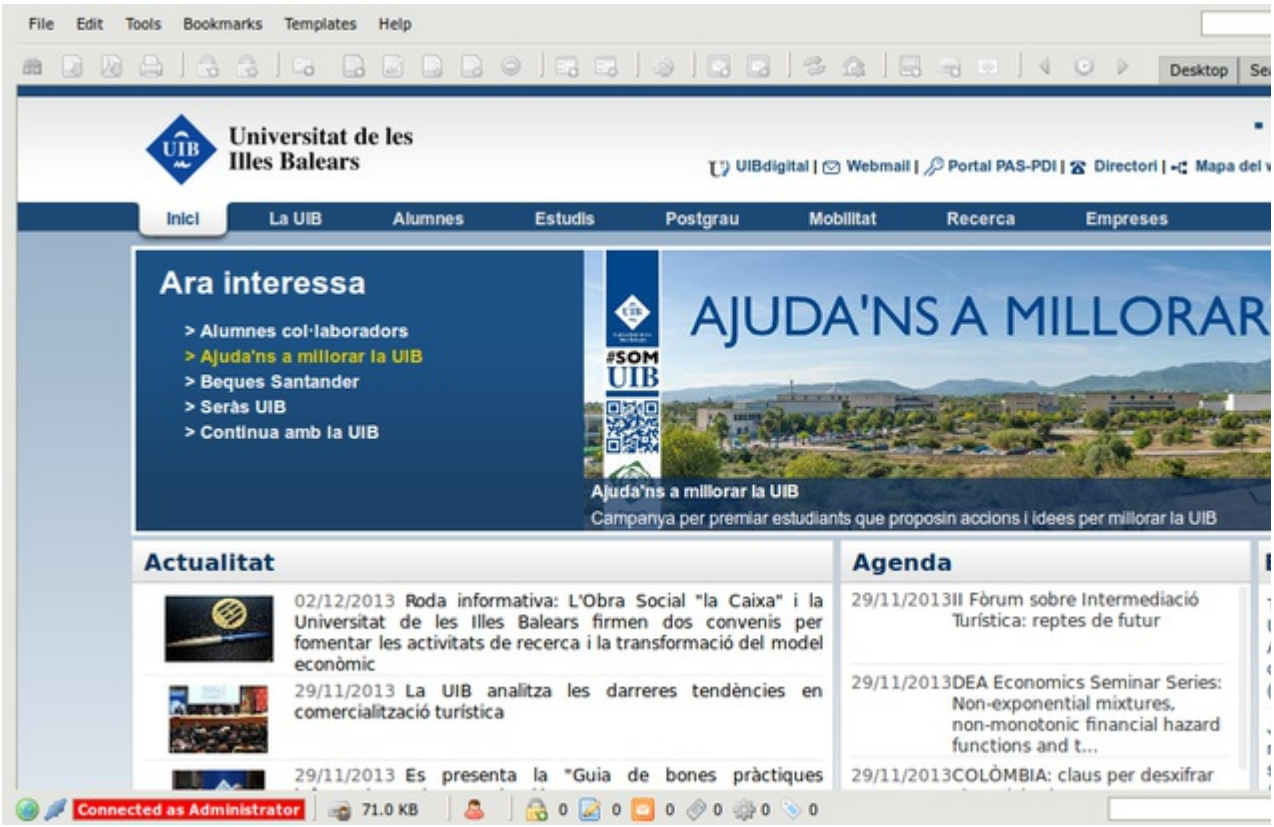
Configuration

Field / Property	Type	Description
extension.extra.tab.workspace	**List**	*Title and url pair. You can add single line (where the character ; is the separator between title and url) or several lines.* `OpenKM;http://www.openkm.com` *If **URL does not start with http** it's considered as **OpenKM relative path** and will be merged with your OpenKM server URL like **YOUR_OPENKM_URL/url***
webservices.visible.properties	**List**	*To make accessible to all the users the values of the parameter "**extension.extra.tab.workspace**", must be **added the value** "**extension.extra.tab.workspace**" in the current list.* *If you forget to add the property name, only administrator users will be able to get the value so some extensions may produce errors for regular users.*

More information at Configuration parameters.

Sample

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


File Edit Tools Bookmarks Templates Help
Desktop
Universitat de les Illes Balears
UIBdigital | Webmail | Portal PAS-PDI | Directori | Mapa del
Inici La UIB Alumnes Estudis Postgrau Mobilitat Recerca Empreses
Ara interessa
> Alumnes col·laboradors
> Ajuda'ns a millorar la UIB
> Beques Santander
> Seràs UIB
> Continua amb la UIB
#SOM UIB
AJUDA'NS A MILLORAR
Ajuda'ns a millorar la UIB
Campanya per premiar estudiants que proposin accions i idees per millorar la UIB
Actualitat
02/12/2013 Roda informativa: L'Obra Social "la Caixa" i la Universitat de les Illes Balears firmen dos convenis per fomentar les activitats de recerca i la transformació del model econòmic
29/11/2013 La UIB analitza les darreres tendències en comercialització turística
29/11/2013 Es presenta la "Guia de bones pràctiques
Agenda
29/11/2013 II Fòrum sobre Intermediació Turística: reptes de futur
29/11/2013 DEA Economics Seminar Series: Non-exponential mixtures, non-monotonic financial hazard functions and t...
29/11/2013 COLÒMBIA: claus per desxifrar
Connected as Administrator
71.0 KB

External preview extension

The External Preview feature helps you to preview documents based on external URL request.

*This option is only available if you have enabled the **External preview** feature.*

There are two parameters passed as request parameter in URL request:

- *The document **uuid**.*
- *The **downloadUrl**.*

Configuration

Field / Property	Type	Description

extension.external.preview	List	*Each row couple of values is a preview content declaration, the format is:*
		mime-type;url
		`application/illustrator;http://localhost:8180/openkm/test`
		`application/epub+zip;http://localhost:8180/openkm/epub_pr`
		`application/x-mspublisher;http://localhost:8180/openkm/mspreviewer_prev`

More information at Configuration parameters.

Sample

Configuration

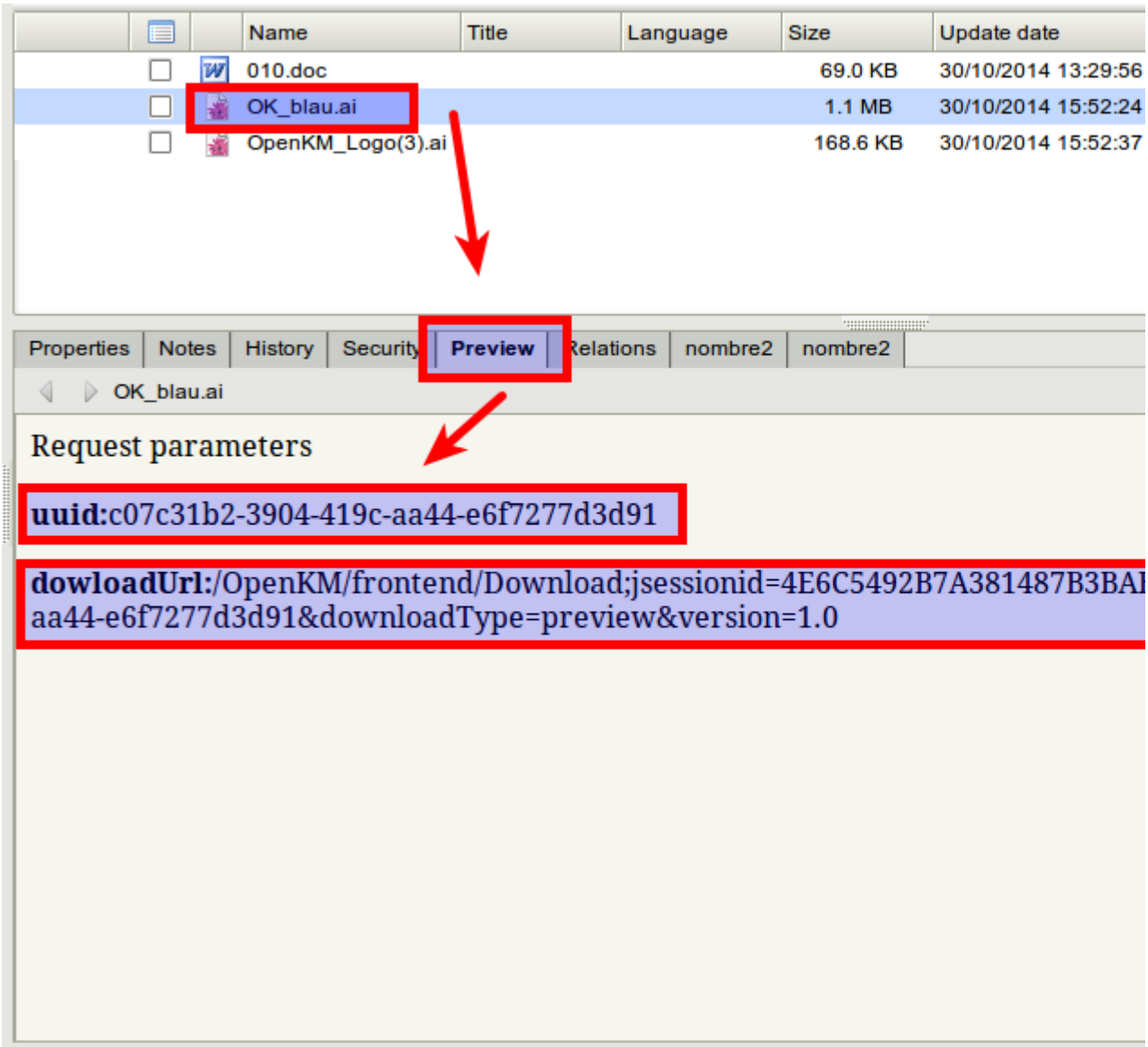
Create file named test.jsp and copy into $TOMCAT_HOME/webapps/openkm folder.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>External URL request example</title>
</head>
<body>
 Request parameters
 <br />
 <br />
 <b>uuid:</b><%=request.getParameter("uuid")%><br />
 <br />
```

```
  <b>dowloadUrl:</b><%=request.getParameter("dowloadUrl")%><br />
 </body>
 </html>
```

Usage

- *Choose a document with one of the MIME type set in configuration parameters.*
- *Show preview tab. Will be shown the external URL based on document mime-type.*



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

External tab document extension

The External Tab Document feature helps you to show tab content in a properties tab based on external URL request when a node document is selected.

*This option is only available if you have enabled the **External tab document** feature.*

There are two parameters passed as request parameter in URL request:

- *The document **uuid**.*
- *The **downloadUrl**.*

Configuration

Field / Property	Type	Description

extension.external.tab.document	**List**	*Each row is a couple of values with a tab content declaration, the format is:* **name;url**
		`name1;http://localhost:8180/openkm/test.jsp`
		`name2;http://www.openkm.com`

webservices.visible.properties	**List**	*To make accessible to all the users the values of the parameter "**extension.external.tab.document**", must be **added the value "extension.external.tab.document"** in the current list.* *If you forget to add the property name, only administrator users will be able to get the value so some extensions may produce errors for regular users.*

More information at Configuration parameters.

Sample

Configuration

Create file named test.jsp and copy into $TOMCAT_HOME/webapps/openkm folder.

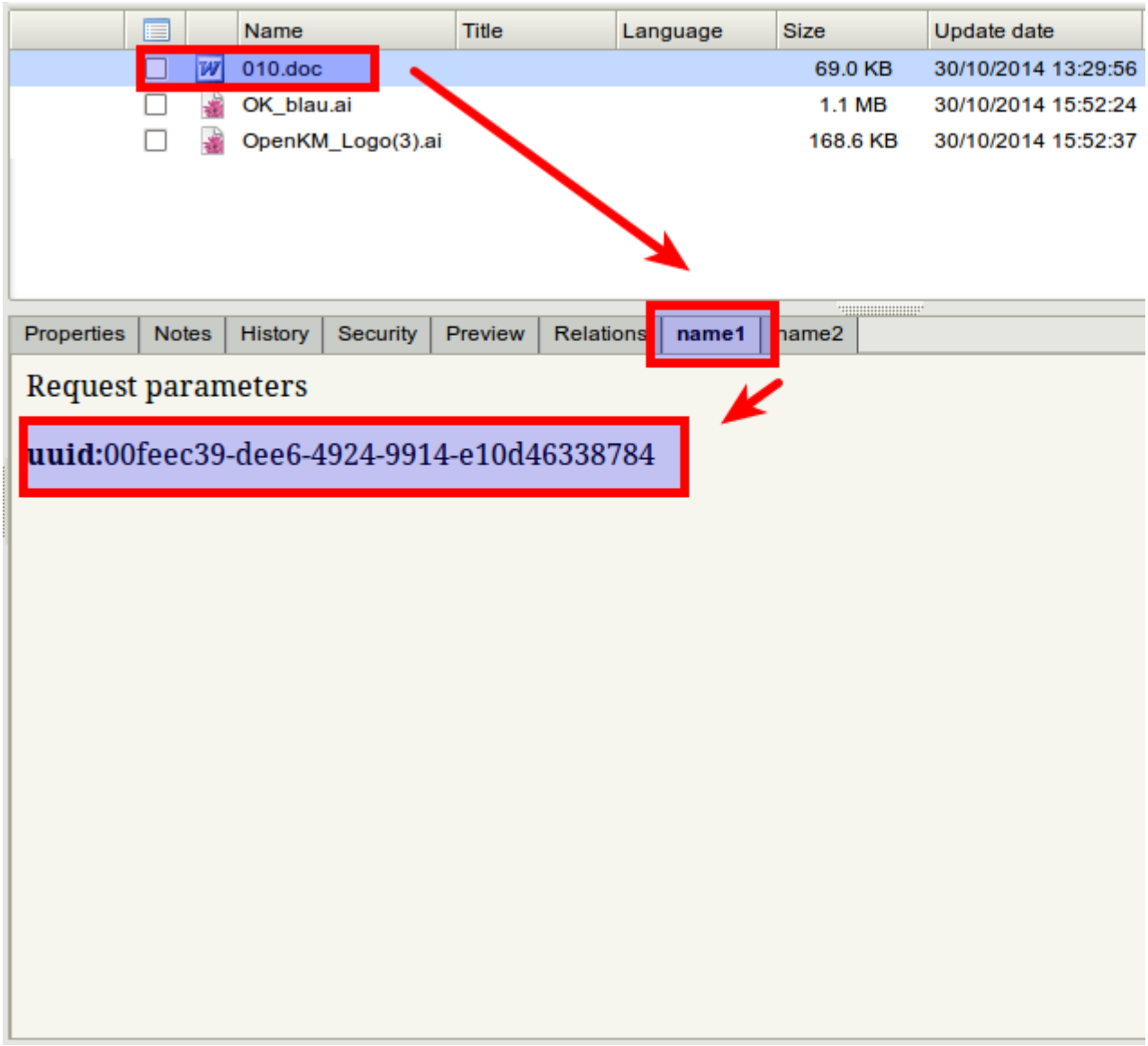
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>External URL request example</title>
</head>
<body>
 Request parameters
 <br />
 <br />
 <b>uuid:</b><%=request.getParameter("uuid")%><br />
</body>
</html>
```

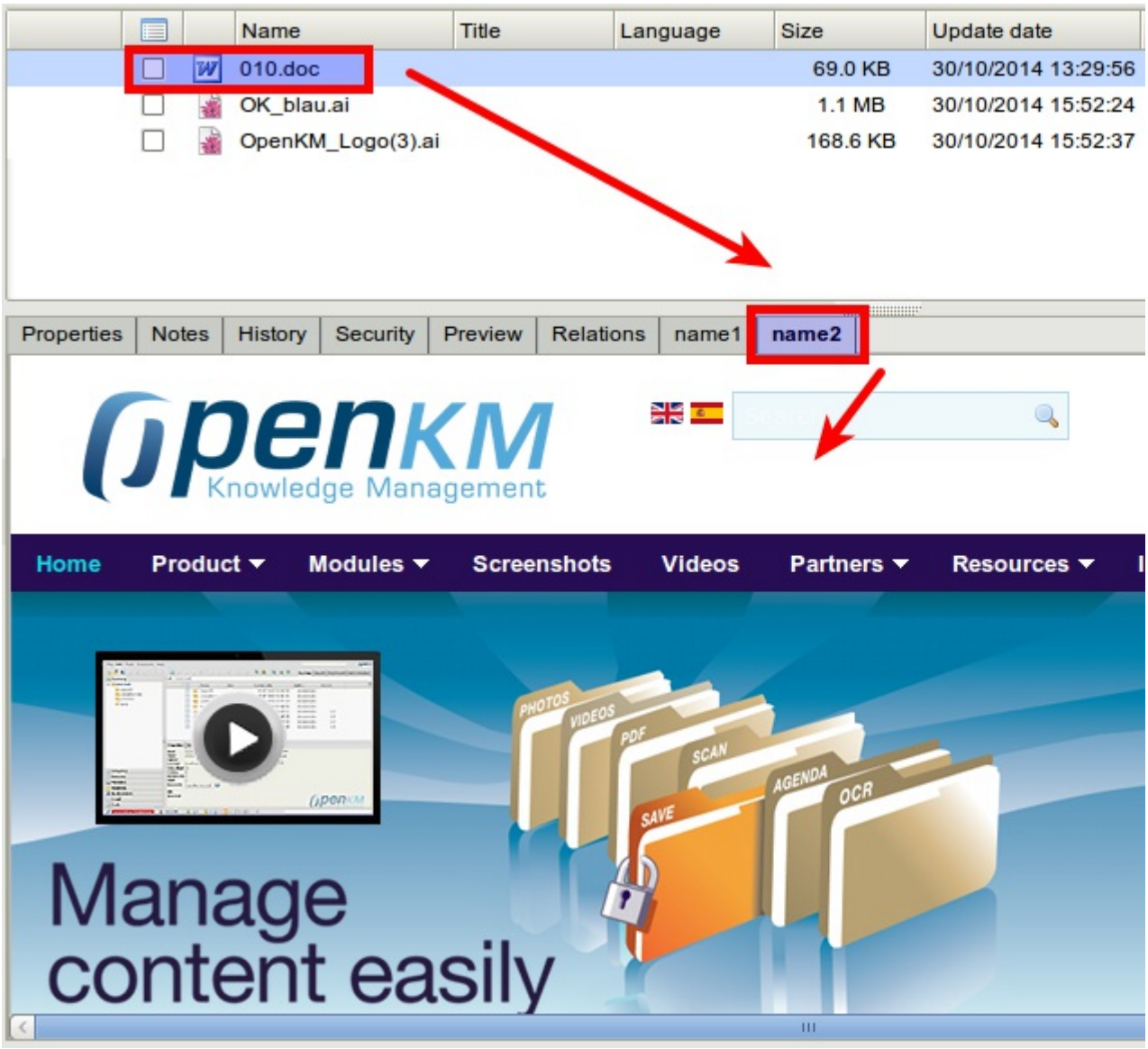
Usage

- ***Choose a document***.
- ***Click on "name1" tab***.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Click on "name2" tab.***



External tab folder extension

The External Tab Folder feature helps you to show tab content in properties tab based on external URL request when a node folder is selected.

*This option is only available if you have enabled the **External tab folder** feature.*

There are two parameters passed as request parameter in URL request:

- *The document **uuid**.*
- *The **downloadUrl**.*

Configuration

Field / Property	Type	Description

extension.external.tab.folder	**List**	*Each row is a couple of values with a tab content declaration, the format is:* **name;url**
		`name1;http://localhost:8180/openkm/test.jsp`
		`name2;http://www.openkm.com`

webservices.visible.properties	**List**	*To make accessible to all the users the values of the parameter "**extension.external.tab.folder**", must be **added the value "extension.external.tab.folder"** in the current list.* *If you forget to add the property name, only administrator users will be able to get the value so some extensions may produce errors for regular users.*

More information at Configuration parameters.

Sample

Configuration

Create file named test.jsp and copy into $TOMCAT_HOME/webapps/openkm folder.

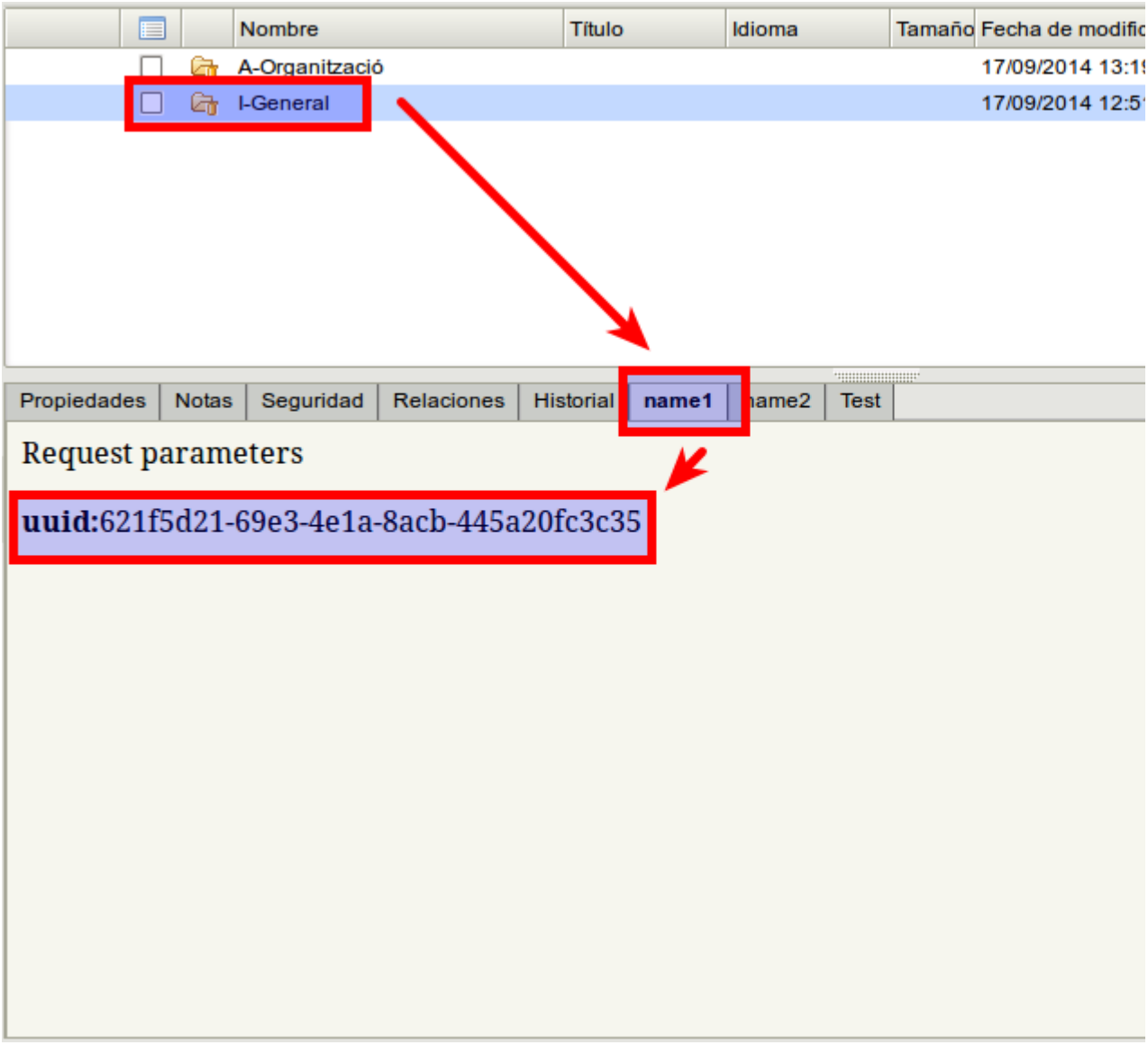
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

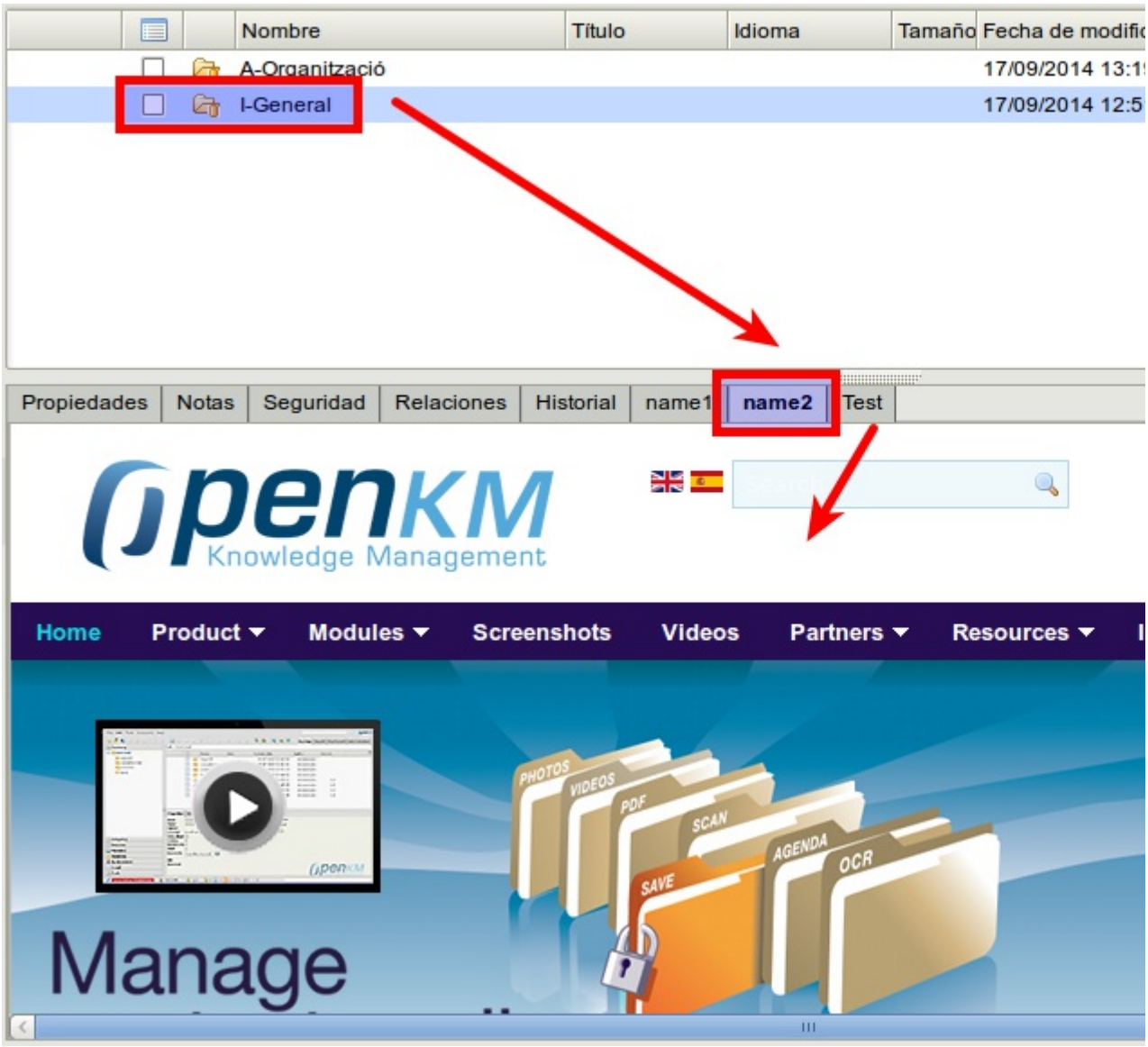
```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>External URL request example</title>
</head>
<body>
 Request parameters
 <br />
 <br />
 <b>uuid:</b><%=request.getParameter("uuid")%><br />
</body>
</html>
```

Usage

- ***Choose a folder***.
- ***Click on "name1" tab***.



- ***Click on "name2" tab***.



External tab mail extension

The External Tab Mail feature helps you to show a tab content in the properties tab based on external URL request when a node mail is selected.

*This option is only available if you have enabled the **External tab mail** feature.*

There are two parameters passed as request parameter in URL request:

- *The document **uuid**.*
- *The **downloadUrl**.*

Configuration

Field / Property	Type	Description

extension.external.tab.mail	**List**	*Each row is a couple of values with a tab content declaration, the format is:* **name;url**
		`name1;http://localhost:8180/openkm/test.jsp`
		`name2;http://www.openkm.com`

webservices.visible.properties	**List**	*To make accessible to all the users the values of the parameter "**extension.external.tab.mail**", must be **added the value** "**extension.external.tab.mail**" in the current list.* *If you forget to add the property name, only administrator users will be able to get the value so some extensions may produce errors for regular users.*

More information at Configuration parameters.

Sample

Configuration

Create file named test.jsp and copy into $TOMCAT_HOME/webapps/openkm folder.

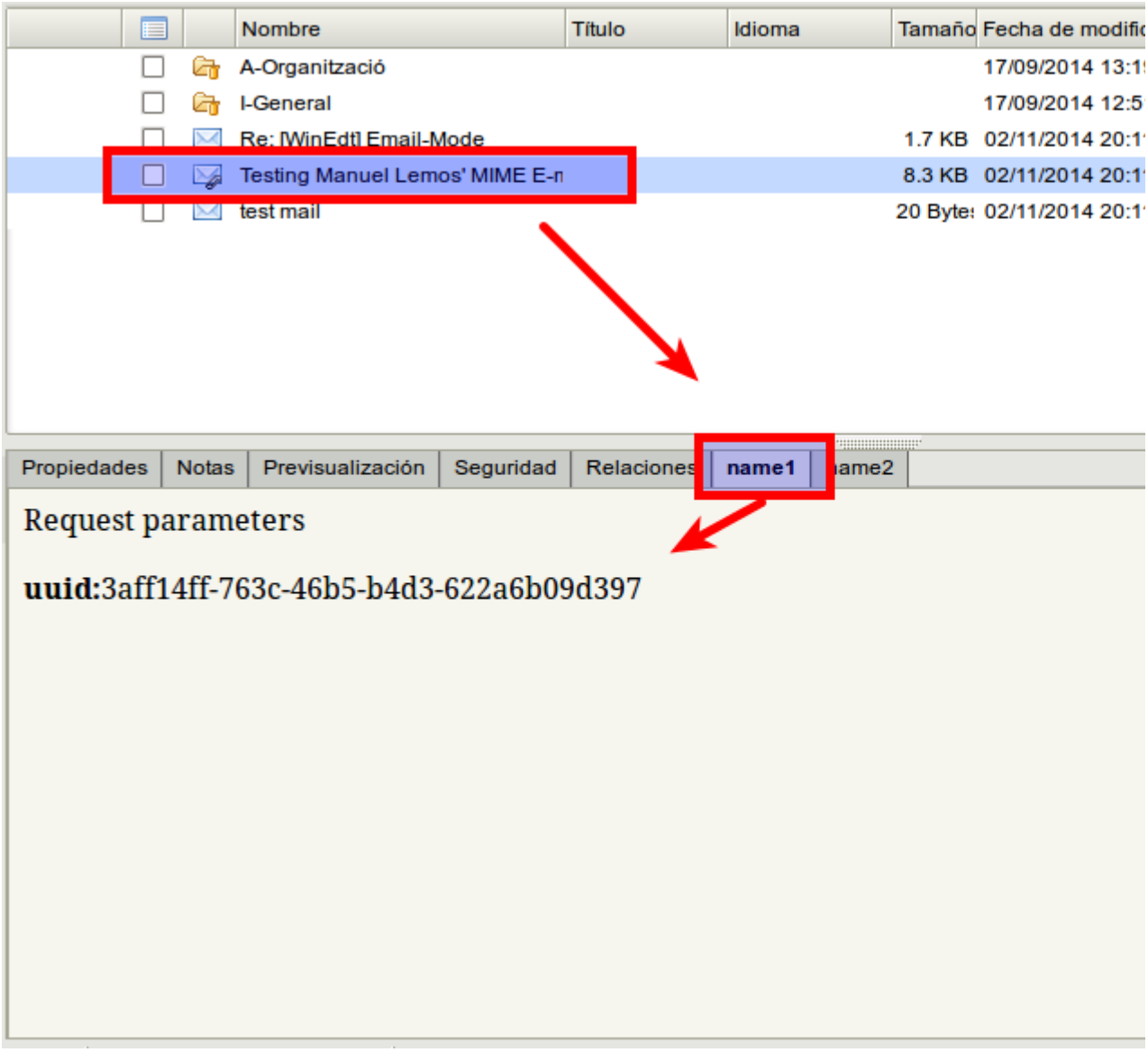
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

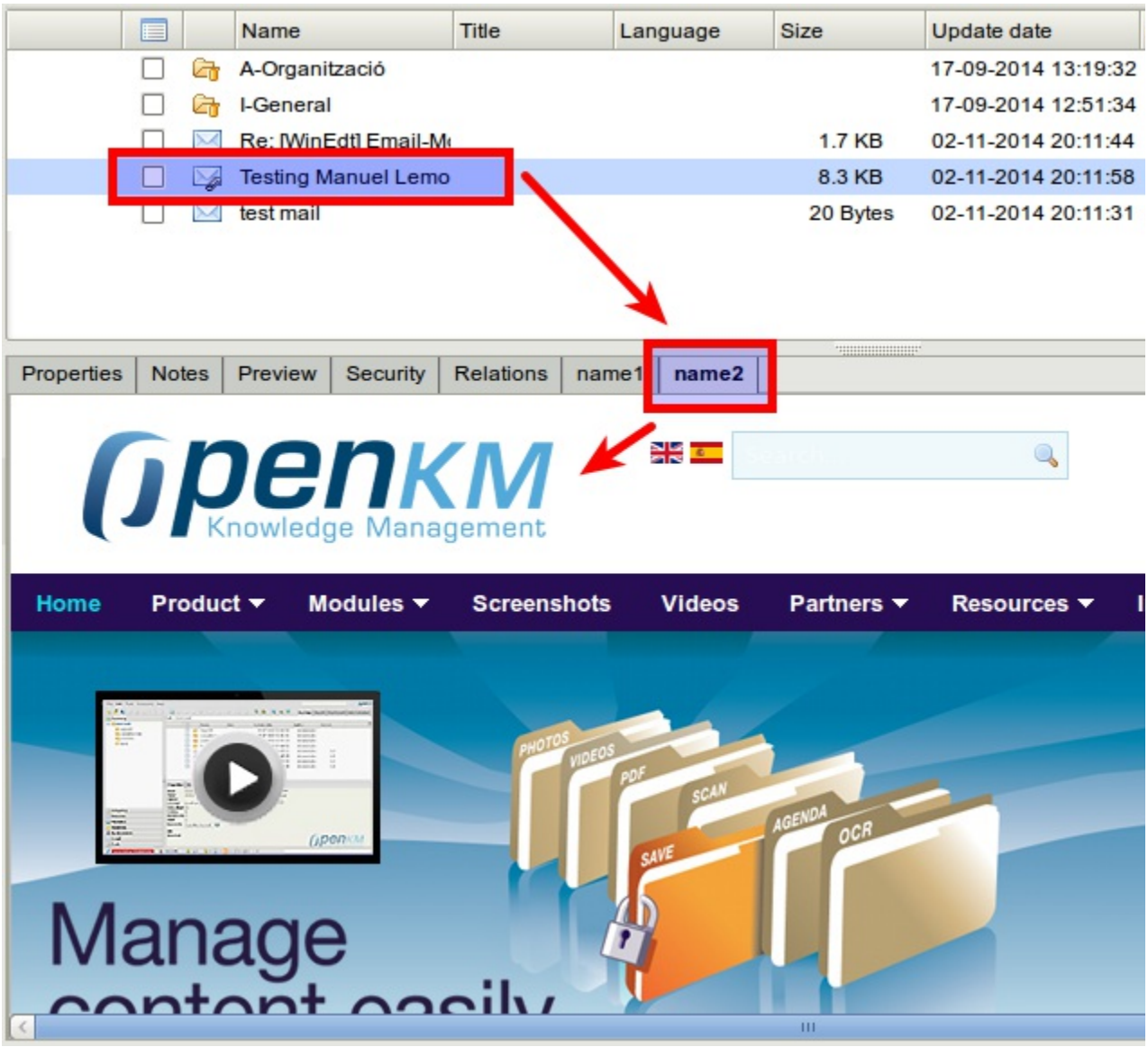
```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>External URL request example</title>
</head>
<body>
 Request parameters
 <br />
 <br />
 <b>uuid:</b><%=request.getParameter("uuid")%><br />
</body>
</html>
```

Usage

- ***Choose a mail***.
- ***Click on "name1" tab***.



- ***Click on "name2" tab.***



External tab record extension

The External Tab Record feature helps you to show a tab content in properties tab based on external URL request when a node record is selected.

*This option is only available if you have enabled the **External tab record** feature.*

There are two parameters passed as request parameter in URL request:

- *The document **uuid**.*
- *The **downloadUrl**.*

Configuration

Field / Property	Type	Description

extension.external.tab.record	**List**	*Each row is a couple of values with a tab content declaration, the format is:* **name;url**
		`name1;http://localhost:8180/openkm/test.jsp`
		`name2;http://www.openkm.com`

webservices.visible.properties	**List**	*To make accessible to all the users the values of the parameter "**extension.external.tab.record**", must be **added the value "extension.external.tab.record"** in the current list.* *If you forget to add the property name, only administrator users will be able to get the value so some extensions may produce errors for regular users.*

More information at Configuration parameters.

Sample

Configuration

Create file named test.jsp and copy into $TOMCAT_HOME/webapps/openkm folder.

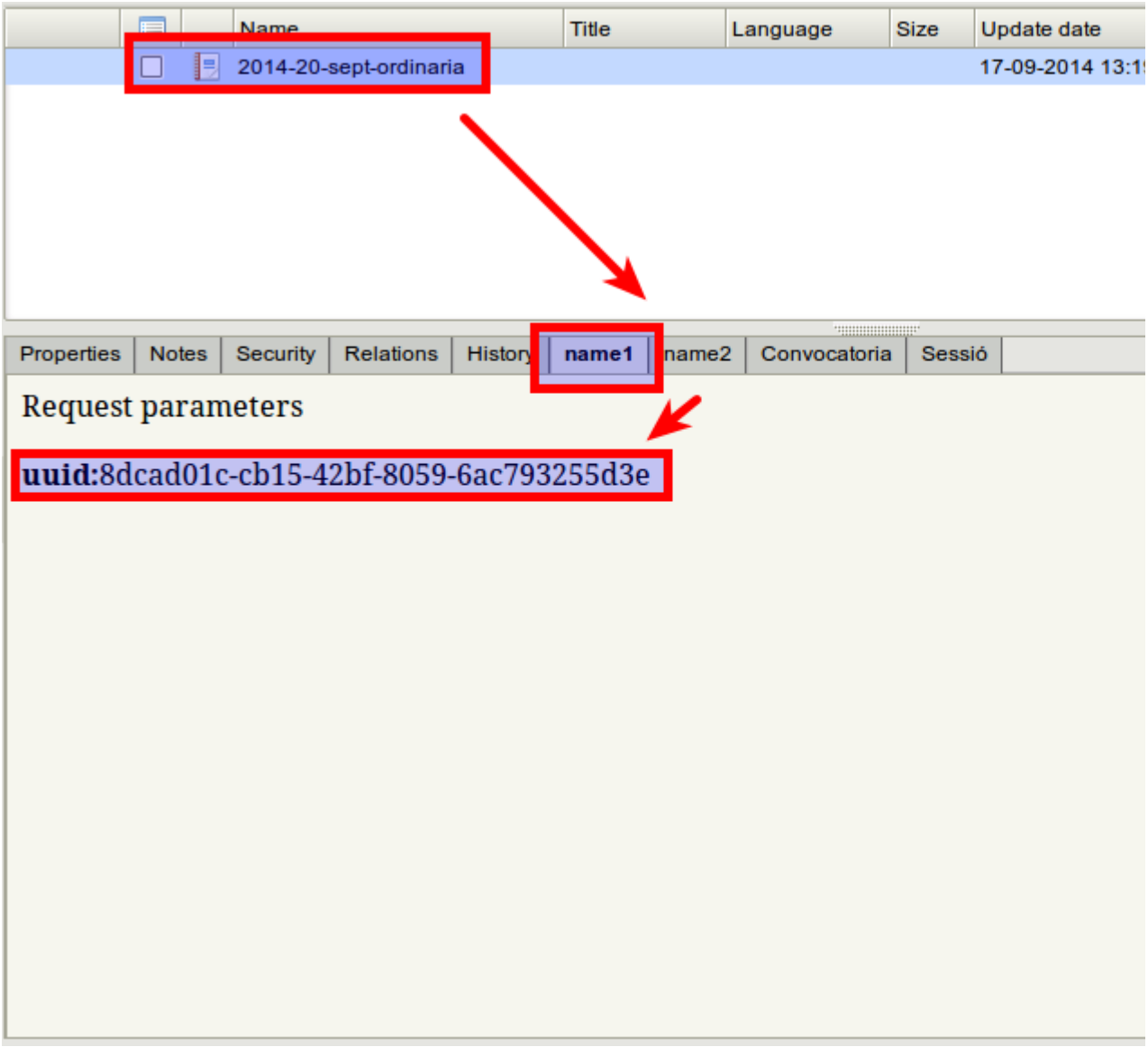
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>External URL request example</title>
</head>
<body>
 Request parameters
 <br />
 <br />
 <b>uuid:</b><%=request.getParameter("uuid")%><br />
</body>
</html>
```

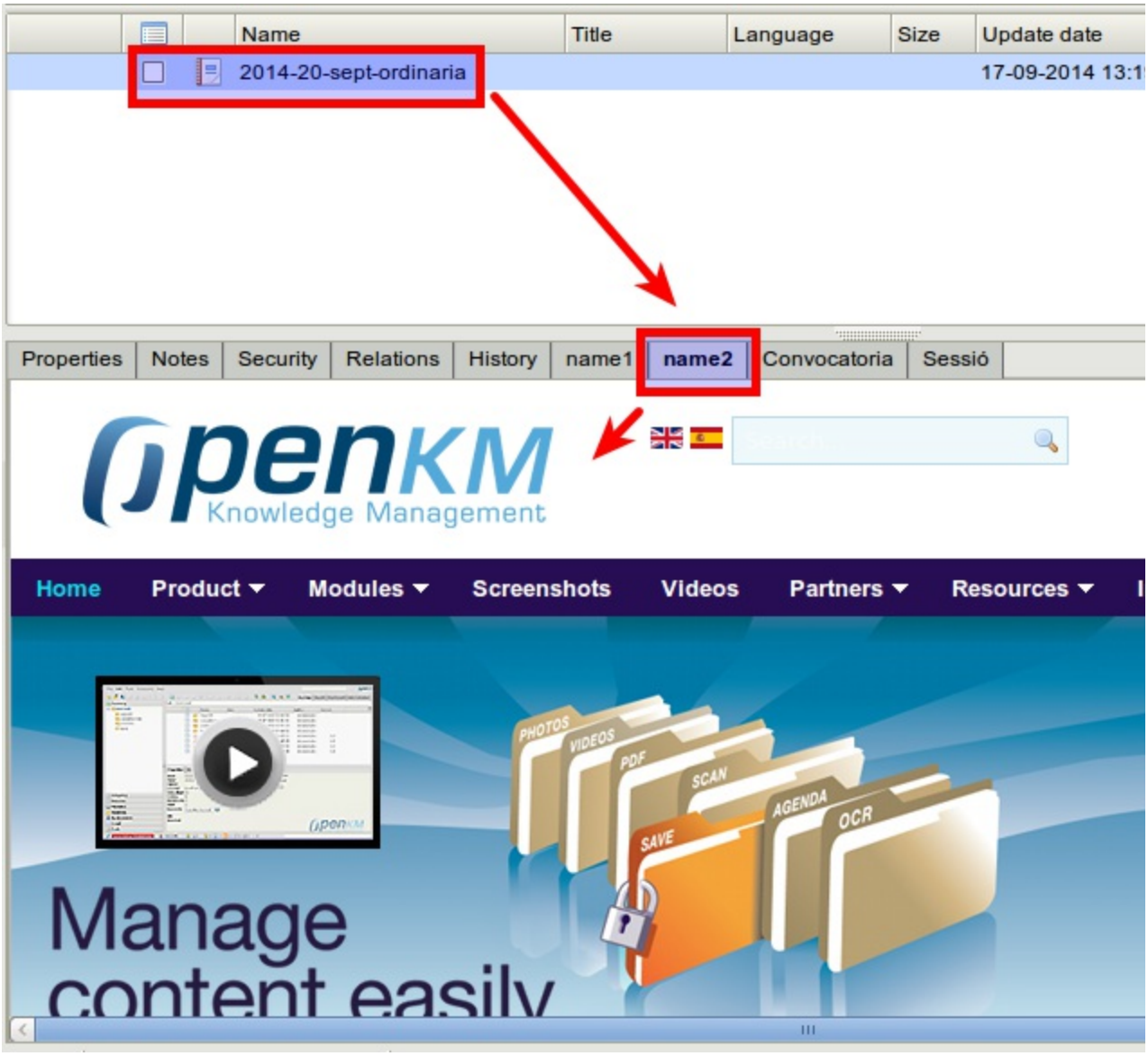
Usage

- ***Choose a record.***
- ***Click on "name1" tab.***



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Click on "name2" tab.***



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Menu options extension

Menu options feature helps you to set new toolbar menu options. Each menu option opens popup with external URL.

*This option is only available if you have enabled the **Menu options** feature.*

There is a parameters passed as request parameter in URL request:

- *The node **uuid** (document, folder, mail or record) .*

Configuration

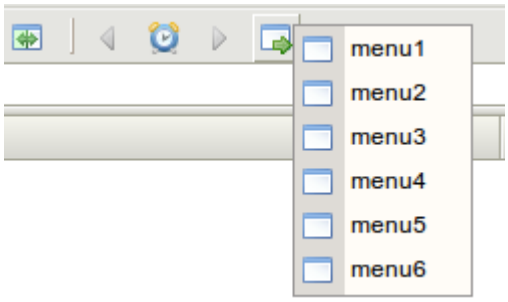
Field / Property	Type	Description
extension.toolbar.menu	**String**	*Each row is a couple of values with a tab content declaration, the format is:* **name;url;width;height**

```
menu1; http://www.openkm.com;500;500
menu2; http://www.openkm.com;700;700
menu3; http://www.wiki.openkm.com;900;300
menu4; http://www.forum.openkm.com;500;900
menu5; http://docs.openkm.com;500;500
menu6; http://issues.openkm.com;700;700
```

More information at Configuration parameters.

Usage

- ***Click on a menu option**. Will be shown a popup with the selected URL.*



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

menu1
OpenKM
Knowledge Management
Home
Product
Modules
Screenshots
Videos
Partners
PHOTOS
VIDEOS
PDF
SCAN
SAVE
AGENDA
OCR
Manage
content easily
TO INSTALL, MANAGE OR SHARE
Cerrar

Forum extension

The Forum extension helps you creating discussion topics on nodes (documents, folders, mails or records). Also there is a general purpose forum available from Dashboard view.

*This option is only available if you have enabled the **Forum** feature.*

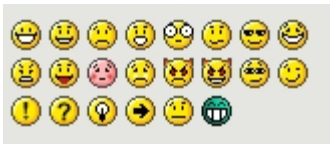
Only administrators can create, update or delete forums from Dashboard view.

Only administrators can view the "document discussions" forum in Dashboard view. This forum contains all the node topics.

Nobody can delete "document discussions" forum in Dashboard view.

Editor features

Feature	Description
B	Bold.
I	Italic.
U	Underline.
p	New paragraph.
Quote	Make a quotation.
Img	Add image url.
Email	Add email.
h1 to h6	Header settings.
Align Left, center, justify and right.	Set text justification.

Text size	*Set text size*
Show colour	*Palette of colors.*
Smilies	*Smilie icons.* 
	Add document link.
	Add image.
	Add folder link.
	Add mail link.
	Add record link.

Create new topic

- ***Choose a node*** *(document, folder, mail or record).*
- *Click on* ***New topic button***.
- ***Fill the form***.
- *Click on* ***Create button***.

Create reply

- ***Choose a topic on a node***.
- *Click on* ***Post reply button***.
- ***Fill the form***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on* ***Create button***.

Update post

- ***Choose a topic on a node***.
- *Click on* ***Edit icon***.
- ***Modify the form***.
- *Click on* ***Update button***.

Delete post

- ***Choose a topic on a node***.
- *Click on* ***Delete icon***.
- *Click on* ***Accept button***.

Create forum

> *This option is available in Dashboard view only for administrators .*

- *Click on* ***New forum button***.
- ***Fill the form***.
- *Click on* ***Create button***.

Update forum

> *This option is available in Dashboard view only for administrators .*

- *Click on* ***Edit icon***.
- ***Modify the form***.
- *Click on* ***Update button***.

Delete forum

> *This option is available in Dashboard view only for administrators .*

- *Click on* ***Delete icon***.
- *Click on* ***Accept button***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Stamp extension

The Stamp feature helps you to insert an image into a document.

*This option is only available if you have enabled the **Stamp** feature.*

You can stamp a document from image or text.

Document stamping

- ***Choose a document** (PDF or convertible to PDF).*
- *Click on* ***Stamp icon**.*
- ***Choose the stamp type** from list.*
- *Click on **Stamp button**.*

When selected document is not PDF, it will be created a new PDF document with stamp into.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Using OpenKM from external applications

There are a number of options available where you can access and use OpenKM from external applications.

Microsoft Office Add-in

You can create, edit and save files that are stored in the OpenKM repository, directly from Microsoft Office applications such as Microsoft Word, Microsoft Excel, Microsoft Power Point and Microsoft Outlook.

With OpenKM Microsoft Outlook Addin you can save mails to OpenKM repository from within Microsoft Outlook.

More information at Microsoft Office Add-in.

Desktop sync

OpenKM Desktop Sync synchronizes files between a local desktop and OpenKM repository using SDK for Java.

More information at Desktop Sync.

Digital Signature Client

Digital Signature Clients help you to quickly sign - electronic digital signature - documents to OpenKM.

More information at Digital signature client.

Scanner client

Scanner Client allows to capture images from scanners and upload them to the OpenKM repository.

Optionally:

- *The images can be convert to PDF file.*
- *Can add metadata group while uploading files to the OpenKM repository.*

More information at Scanner client.

Desktop Importer tool

The Desktop Importer tool is designed to import documents to a remote OpenKM instance. It uses OpenKM WebServices API so is firewall friendly.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

More information at Desktop importer tool.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Microsoft Office Add-in

You can create, edit and save files that are stored in the OpenKM repository, directly from Microsoft Office applications such as Microsoft Word, Microsoft Excel, Microsoft Power Point and Microsoft Outlook.

With OpenKM Microsoft Outlook Add in you can save mails to OpenKM repository from within Microsoft Outlook.

*Importing mail creates a folder structure based on received email date (**year/month/day** folder structure) and imports by default to "**/okm:mail/{userId}**" node.*

*When a user edits a document, they are saved locally in "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM**".*

When editing OpenKM documents, a user can decide to upload or cancel editing. After both operations the local file is removed so as not to generate user confusion. When you upload a new file, it is not removed from your computer.

*In the file "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/openkm.xml**" are saved the connection OpenKM data. The password is crypted.*

*In the file "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/document.xml**" are saved the documents edition information status.*

There is a log file for each application at:

- *"**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/WordAddin.log**".*
- *"**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/ExcelAddin.log**".*
- *"**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/PowerPointAddin.log**".*
- *"**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/OutlookAddin.log**".*

*Microsoft Office Add in is **compatible with versions 2000 to 2013**.*

Configuring add-in

- *Click on configuration toolbar icon (menu option at Microsoft Outlook).*
- *Set version to "**Webservices 2.0**".*
- *Choose language (by default English).*
- *Set the **username**.*
- *Set the **password**.*
- *Set the **host** (for example http://localhost:8080/OpenKM).*
- *Click on **Test** button to check the connection configuration.*

Editing documents from Microsoft Office

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Depending your Microsoft Office version you will see an OpenKM ***toolbar*** *or a* ***Ribbon****.*

Feature	*Description*
OpenKM ***Configuration***	*Configure the connection to OpenKM server.*
Add document	*Upload a new document into the OpenKM repository.*
Edit	*Edit document (check-out).* *When a user edits a document, they are saved locally in "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM**".* *You can continue editing the document other day (the document status is saved at "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/document.xml**" file.*
Update	*Upload a new document version (check-in).*
Cancel edit	*Cancel document edition.*
Help	*OpenKM to OpenKM Microsoft Office Add in URL in your default web browser.*

Importing mails from Microsoft Outlook

- ***Choose*** *one or several* ***mails****.*
- ***Click on import*** *menu option.*
- ***Follow the wizard*** *to select the destination folder. When grouped checkbox is enabled will be created the* ***year/month/day*** *folder structure.*
- *Click on* ***Upload*** *button.*

Add new translation

*The translation file is set at "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/OKMAddin-enGB.properties**". The easiest way is to change the contents of the file directly.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Create a new translation file at "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/**" (for example copy OKMAddin-enGB.properties to OKMAddin-esES.properties for Spanish translation).*
- *Set the translations.*
- *Edit "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/openkm.xml**" and change the tag*

```
<idioma>enGB</idioma>
```

to

```
<idioma>esES</idioma>
```

Know issues

In Microsoft Office 2010 we have detected some issue, if you detect some problem using this version try it:

- *Open the menu **File > Options**.*
- *Select **Trust Center** and click on "**Trust Center Settings**" button.*
- *Open **Protected View** option and **disable the first three** checkbox.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Digital Signature Client

Digital Signature Clients help you to quickly sign - electronic digital signature - documents to OpenKM.

*In the file "**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/openkm.xml**" are saved the connection OpenKM data. The password is crypted.*

*The Digital Signature Client takes advantage of using **Microsoft Digital Certificates stores**.*

There is a log file for each application at:

- *"**%USER_LOCAL_DOCUMENT_FOLDER%/OpenKM/DigitalSignature.log**".*

Configuring

- *Click on **File > Configuration**.*
- *Set version to "**Webservices 2.0**".*
- *Choose language (by default English).*
- *Set the **username**.*
- *Set the **password**.*
- *Set the **host** (for example http://localhost:8080/openkm).*
- *Click on **Test** button to check the connection configuration.*

Sign documents

- *To sign documents, **select files or folders** with files into and **drag to Digital Signature Client**.*
- *Will be shown a Pop-up to choose the certificate to be used to sign documents. **Select a certificate** and click the **Accept** button.*
- ***Select the destination folder** to save the documents and click **Sign** button.*

If Digital Signature Client detects an existing document with the same name in the repository, it will show a warning asking to update or skipp the document.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Scanner client

Scanner Client allows to capture images from scanners and upload to the OpenKM repository.

Optionally:

- *The images can be converted to PDF file.*
- *Can add metadata group while uploading files to the OpenKM repository.*

Feature		Description
	Scan	*Starts scanning process.*
	Convert to PDF	*Convert an image or group of selected images to a PDF document.*
	Refresh	*Refreshes tree.*
	Upload	*Uploads a document.*
	Delete PDF page.	*Deletes a page from the selected PDF document.*
	Add page to PDF	*Adds new page to the selected PDF document.*

Requirements

Before running the application make sure you have installed:

- *Visual C++ Redistributable for Visual Studio*
  - *.NET 4.0: Visual C++ Redistributable for Visual Studio 2012 (x86 or x64) (Recommended).*
  - *.NET 2.0: Visual C++ Redistributable for Visual Studio 2008 (x86 or x64).*
- *PDF Acrobat Reader.*
- *Microsoft .Net Framework 3.5 or upper.*

Configuring

- *Click on **File** > **Configuration**.*
- *Set version to "**Webservices 2.0**".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Choose language (by default English).*
- *Set the **username**.*
- *Set the **password**.*
- *Set the **host** (for example http://localhost:8080/OpenKM).*
- *Click on **Test** button to check the connection configuration.*

Compression CCITT Group4

- *Click on **Configuration** > **Compression group4** menu option.*
- *Check the **black and white CCITT Group4 compression** checkbox*
- *Set the **percentage** (Thereshold), by default is set to 80%.*
- *Click on **Accept** button.*

Setting destination folder

- *Click top right **Change** button.*
- *Select the destination folder and click on **Select** button.*

Setting scanner configuration

- *Click on Configuration / **Scanner configuration**.*
- *Change the configuration parameters.*
- *Click on **Accept** button.*

We suggest TWAIN protocol rather WIA for better scanner features support.

Scanning

- *Click on **Scanning** button.*
- *Follow the wizard.*

*Will be created a folder based on date (**year-month-day**) into your server at **My images/OKMScan/Images**.*

Convert scanned images to PDF

- ***Choose one or several image files** from tree.*
- *Click on **Images to PDF** button.*

*A PDF file will be created into a folder based on date (**year-month-day**) into your server at **My images/OKMScan/Pdfs.***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

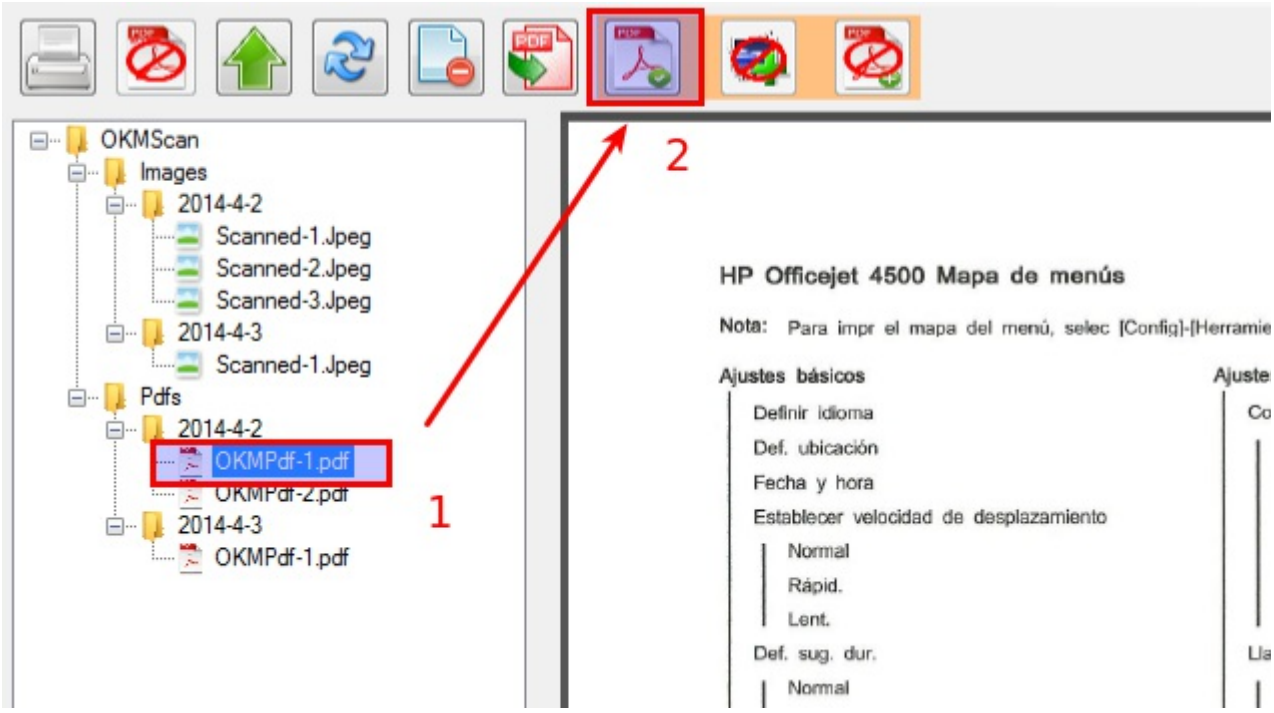
Uploading

- ***Choose one or several files*** *(images or PDF).*
- *Click on* ***Upload*** *button.*
- ***Optionally you can set metadata groups***.
    - ***Choose the metadata group***.
    - *Click on* ***Add*** *button.*
    - ***Fill the metadata group fields***.
    - *Click on* ***Add*** *button.*

Add images to an existing PDF file

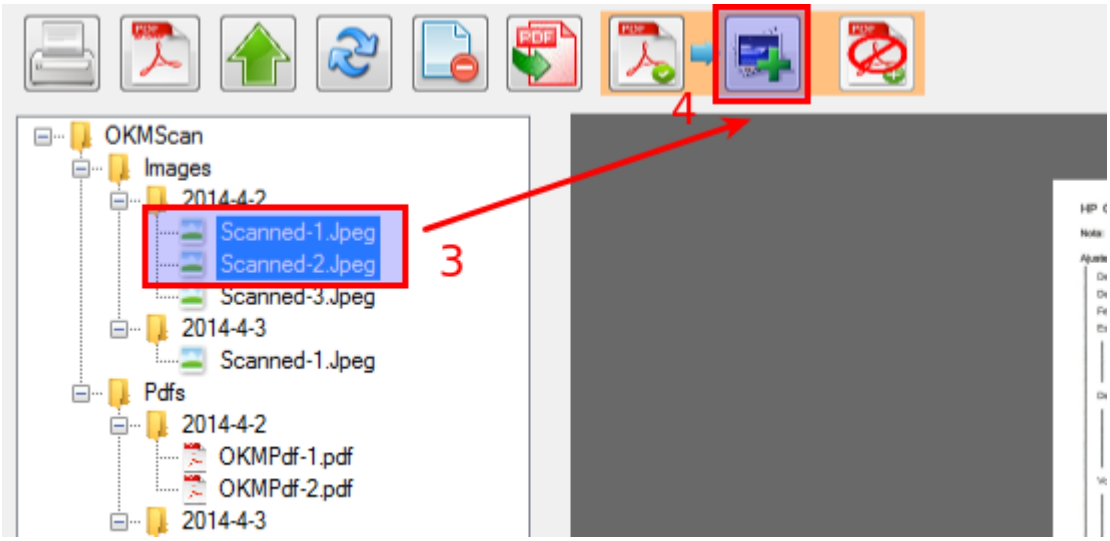
- *Click on* ***Add page to PDF*** *button. Will be shown extra icons in the toolbar.*

- *Action has tree steps, first is to* ***select a PDF file*** *and then click on the icon* *to* ***confirm PDF selection***.



- *The second step is to* ***select one or several images*** *and click on the icon* *to* ***confirm images selection***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



- *And finally click on the icon* *to* ***add the images****. Will be shown a popup to confirm the action, click on* ***Yes*** *button.*

Remove pages from existing PDF file

- ***Choose a PDF file****.*
- *Click on* ***Delete PDF page*** *button.*
- ***Select a range*** *of pages.*
- *Click on* ***Accept*** *button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Desktop Sync

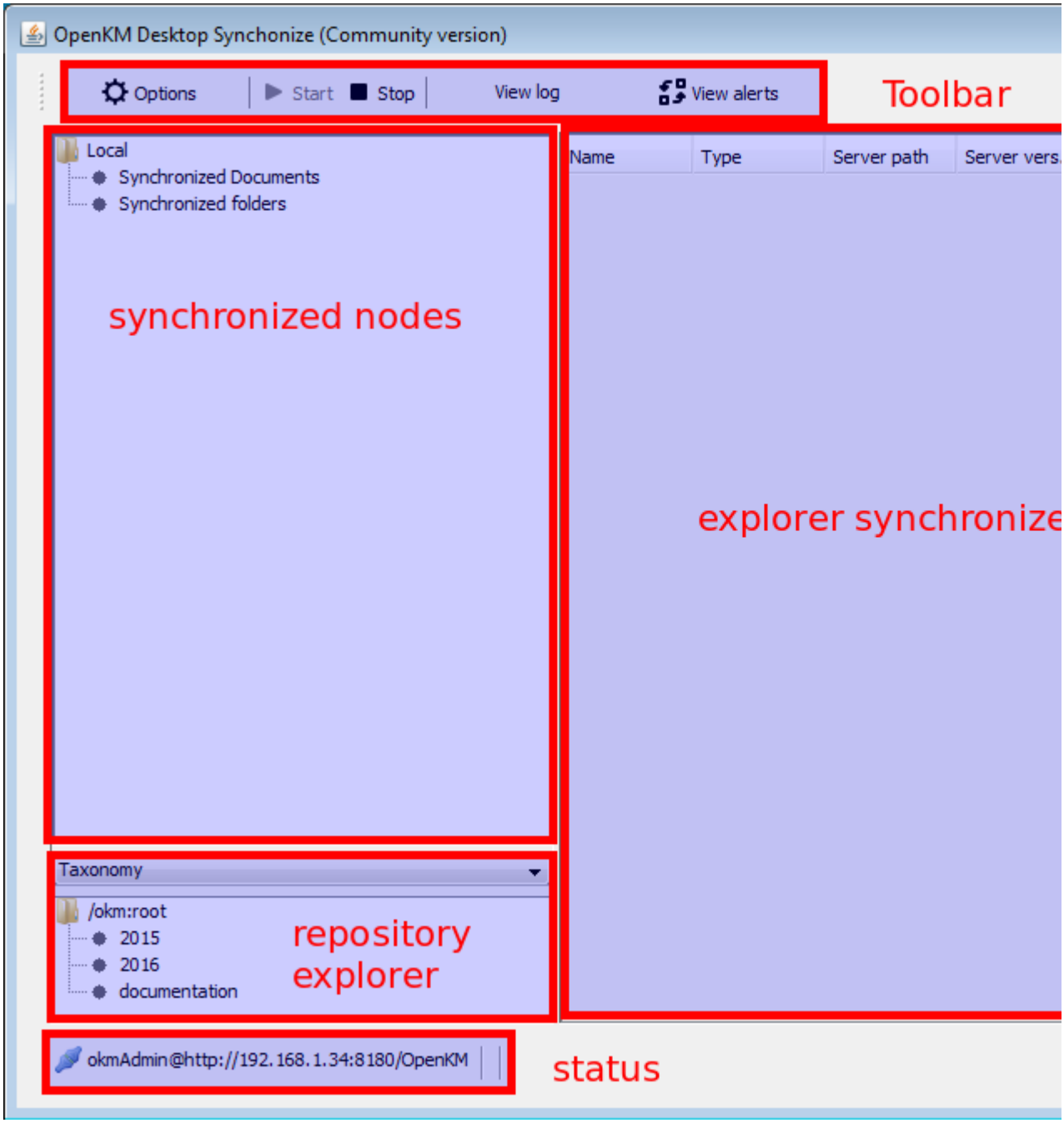
The OpenKM Desktop Sync client allows desktop users to synchronize folders and files between OpenKM server and local computer.

> *This application can synchronize a folder or a single document.*

There are several sections:

- *Toolbar.*
- *Synchronized nodes.*
- *Explore synchronized node.*
- *Repository explorer.*
- *Status.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Section	*Description*
Toolbar	*From toolbar panel you can:* • *Change configuration with **Options** button.* • *Start synchronization with **Start** button.* • *Stop synchronization with **Stop** button.* • *View audit trail with **View log** button.*

	• *Display application alerts with **View alerts** button.*
Synchronized nodes	*From synchronized nodes you can view the folders and documents you are synchronizing.*
Explore synchronized nodes	*In this panel you will see all actions done, filtered by the node you have selected on Synchronized nodes panel. For example a document modified locally that is uploaded to the repository, etc.*
Repository explorer	*From this panel you can navigate across the repository and choose a node - folder or document - to be synchronized.*
Status	*From this panel you can see the connection status and the application notification alerts are also shown.*

Limitations of the application

- *Changes made locally on synchronized folders or documents while the application is stopped will not be take any effect.*

For performance reasons, the synchronization mechanism is based on filesystem listener, when application is stopped the listener obviously is stopped. Although in this scenario we could apply some heuristic based on modification date, for performance reason at the present we are not doing. Take in consideration that any change done while application is stopped will be transparent for the application.

Configuring

The first time you start the application is shown configuration popup:

- *Set the **working path** (file system destination folder).*
- *Set the **host** (for example http://localhost:8080/OpenKM).*
- *Set the **username**.*
- *Set the **password**.*
- *Set the restricted extensions (by default .back,.~,.odesktop). Files with these extensions will not be synchornized.*
- *Set invalid characters (by default the character ":" is not allowed).*
- *Set refreshing time (by default each 60 seconds).*
- *Set your browser (by default "google-chrome". You can change to "firefox" or "iexplorer").*
- *OS explorer (by default "nautilus" for linux. You can change to "explorer" on Windows).*
- *Click on **Test** button to check the connection configuration.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change initial configuration

- *Click on **Systray OpenKM icon > Show desktop > Options**.*
- *Set version to "**Webservices 2.0**".*
- *Choose language (by default English).*
- *Set the **username**.*
- *Set the **password**.*
- *Set the restricted extensions (by default .back,.~,.odesktop). Files with these extensions will not be synchornized.*
- *Set invalid characters (by default the character ":" is not allowed).*
- *Set refreshing time (by default each 60 seconds).*
- *Set your browser (by default "google-chrome". You can change to "firefox" or "iexplorer").*
- *OS explorer (by default "nautilus" for Linux. You can change to "explorer" on Windows).*
- *Click on **Test** button to check the connection configuration.*

Application will create in your server a folder structure:

- *documents (are stored single documents synchronized).*
- *folders (are stored synchronized folders).*
- *log (the application log).*

*The configuration file is saved at "**%UserHome%/.okmDesktopConfiguration**". The password is encrypted.*

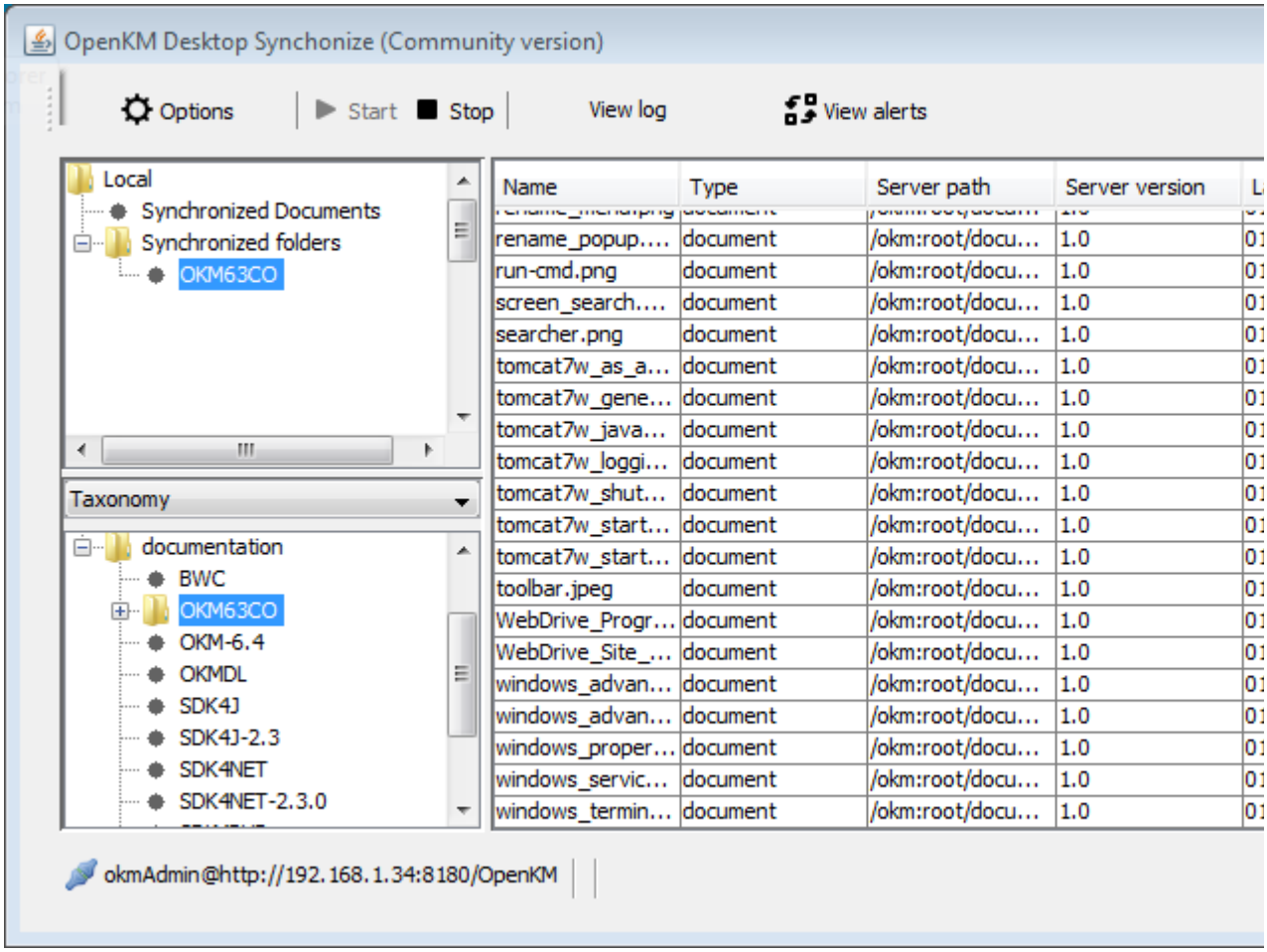
Synchronizing

- *From the Repository explorer panel **choose some node** (folder or document).*
- ***Right click on mouse** and it will be shown a contextual menu.*
- ***Click on Synchronize folder** menu option (in case you selected document the menu option is named Synchronize document).*

At the bottom right you will see the progress of the proces.

When finished choose the synchronized node from Synchronized nodes panel and you'll see the information of all the folders and documents in that node:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Remove synchronized node

- ***Choose the node*** *from Synchronized nodes panel.*
- ***Right click*** *on mouse will be shown a contextual menu.*
- *Click on* ***remove*** *menu option.*

Resolving conflict

The most common conflict happens when you're editing a document and at the same time it has been modified in the OpenKM repository. That means you have modified older version than is now present on the repository. In this case the application will not upload your local version and will be declared as CONFLICT. You must solve it manually :

- ***Backup your document local version***.
- ***Choose the document*** *from the Explore synchronized node.*
- ***Right click*** *on mouse will be shown a contextual menu.*
- *Click on* ***resolve conflict*** *menu option.*
- *Choose* ***Update from server*** *action.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Click on **Resolve** button.*

*If you choose "**Update from server action**", the remove file will overwritten your local file, and you will lose your changes. The idea is then get your document backup and merge manually.*

*Also there's an action named "**Update from locally**" what upload local version to OpenKM server. We suggest use this action with care, because other user has modified the document at the OpenKM repository and probably this changes are not present in your document.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

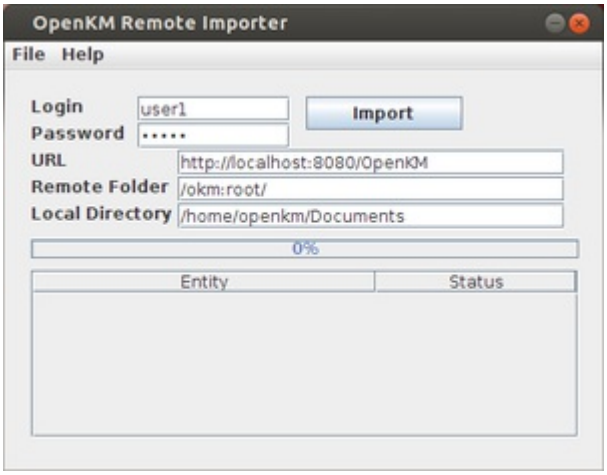
Desktop importer tool

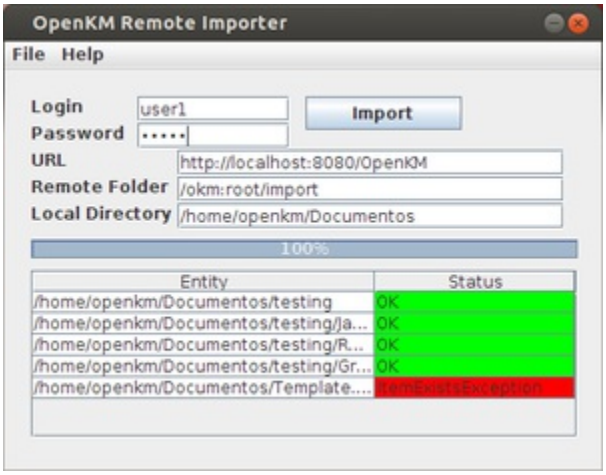
This utility is designed to import documents to a remote OpenKM instance. It uses OpenKM WebServices API so is firewall friendly.

You need Java 7 or upper installed in your computer.

You can directly download and execute the desktop-importer.jnlp file from the web.

Images





Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Working with PDF tools

> *PDF tools are available in OpenKM from OpenKM version 7.1.6.*
>
> *It comes enabled by default in the OpenKM KCenter user interface from version 7.1.18.*
>
> *In the OpenKM classic user interface must be enabled in combination with External tab document extension.*

PDF tools are a set of features what allow to transform PDF files. The PDF transform features are:

Feature	Description
Split	*Split the chosen pages in a single file for each one.*
Rotate	*Rotate the chosen pages in a new single file.*
Extract	*Extract the chosen pages in a new single file.*
Remove	*Remove the chosen pages.*

Steps to enable PDF tools in the classic user interface

- *Enable the extension External tab document extension.*
- *Register the URL in the configuration parameter named "**extension.external.tab.document**".*

Field / Property	Type	Description

extension.external.tab.document	**String**	*Take the next URL as a sample, you should change by your own application URL keeping the "**/openkm/vue/pdf-tools/index.jsp**" part.*
		`http://localhost:8080/openkm/vue/pdf-tools/index.jsp`
		For version 7.1.17 and older use:
		`http://localhost:8080/openkm/pdf-tools/index.jsp`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Migration guide

The Updater utility can be used from version 7.1.4 and upper. For older OpenKM version contact with OpenKM Support.

*Download the latest "**Updater utility**" named OKMUpdater.jar*

- *Please, make a backup before any update.*
- *This utility needs access to **update.openkm.com** and **download.openkm.com** domains.*

Once downloaded, put at $TOMCAT_HOME directory and execute:

```
$ java -jar OKMUpdater.jar -l
```

Follow the wizard steps.

More information at Updater utility.

If you access the Internet using a proxy, just configure it when executing the application:

```
$ java -Dhttp.proxyHost=192.168.0.1 -Dhttp.proxyPort=3128 -jar OKMUpdater.jar -l
```

It is recommended to clear browser cache (prevents "The response could not be deserialized" errors) and Java Plugin cache.

In case you have a Cluster configuration be careful because all the OpenKM instances share the same database, so once you have updated a node the others should be updated without executing the actions so they can be reviewed and executed manually. Ask for support if unsure about which actions should be executed in slave nodes.

```
$ java -jar OKMUpdater.jar -l -m
```

Troubleshooting

In early OpenKM installations the upgrade process may need some previous steps. If you see an error like this one, you have this problem:

To be able to use the updater utility you have to follow these steps:

- *Download **the same version of your current OpenKM instance** from http://download.openkm.com.*
- *Stop OpenKM.*
- *Replace the $TOMCAT_HOME/webapps/OpenKM.war file with the downloaded one.*
- *Delete the $TOMCAT_HOME/webapps/OpenKM directory.*
- *Start OpenKM.*

Once OpenKM is running you can use the updater utility again.

If the problem persist, please open a support ticket at http://support.openkm.com and attach OKMUpdater.log and $TOMCAT_HOME/logs/catalina.log files.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Changelog

OpenKM 7.1 Professional

v7.1.20

- *Release: 2021-02-19*
- *Misc improvements in fileplan*
- *Fixed problem with Lucene query*
- *New mail import method in OKMImport API*
- *Experimental repository database migration tool*
- *Several internal fixes and improvements*
- *Improvements and fixes in KCenter:*
    - *Added new import website popup*
    - *Added OCR data capture in the wizard*
    - *Added logic to force cache reload when a new version is detected*
    - *Improvement in the tab search and navigator of the popup search. Now is marked clearly the selected node*
    - *Improvement in the History panel: now you can rename a node version. Only members of "ROLE_VERSION_ADMIN" allowed*
    - *Improvement in the browser view: administrators are allowed to purge nodes anywhere (no trash)*
    - *Improvement in restore node action from the trash: now a confirmation popup is shown*
    - *Improvement in the browser view. The filter works with the "enter" keyword*
    - *Improvement in the dashboard: You can choose visible options using the profile*
    - *Fix bug in the browser view when filter by empty text*
    - *Fix bug in the browser view when clicking on the refresh icon the data of the selected node must be updated*
    - *Fix bug when root nodes are chosen, some actions (like delete) must be disabled*
    - *Fix bug when mapping select data values in workflow form elements*
    - *Fix bug in the Lucene query feature that stopped working*
    - *Fix bug in the size of thumbnail page popup: now it fits window size*

v7.1.19

- *Release: 2021-01-26*
- *New administration (beta)*
- *HTML can generate also PDF documents*
- *Fixed problem with reports and some fields*
- *Improved search results performance*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *New publis URL to share document preview*
- *Save as PDF implemented in mails*
- *Removed deprecated macro feature*
- *Removed deprecated chat feature*
- *Improvements and fixes in KCenter:*
    - *Improvement in document creation from template: HTML and ODT templates can generate PDF files*
    - *Improvement in metadata panel: Metadata groups are shown in tabs and have been added a new feature to list all metadata groups in a single view*
    - *Improvement in dashboard: Workflow task and pooled task are now paginated*
    - *Improvement in lock and unlock feature: You can lock and unlock any node type*
    - *Improvement in purge node from trash: A confirmation popup is shown*
    - *Improvement in search: Searches with several keywords and categories are now available*
    - *Improvement in upload: Shown icon and destination path*
    - *Improvement in search: Now is about 70% faster*
    - *Improvement in HTML editor: Added bullet lists*
    - *Added task share link*
    - *Added document public preview link in properties tab*
    - *Added message to change the expired password*
    - *Added contextual menus in email panel*
    - *Added upload and download documents element in the workflow forms*
    - *Added comments in node versions*
    - *Added preview in the massive actions*
    - *Added status in record properties panel*
    - *Fix bug with parent-child select: The child has only elements based on the parent value*
    - *Fix bug in workflow button. Ask for submit confirmation*
    - *Fix bug in workflow assign button: Avoid execute twice*
    - *Fix bug in workflow button: Avoid executing twice*
    - *Fix bug in workflow form fields of type "select plugin"*
    - *Fix bug in contextual menu: Delete action was missing*
    - *Fix bug when creating a folder or record: Wizard is refreshed after folder or record creation*
    - *Fix bug in email menu: It's not allowed to rename an email*
    - *Fix bug when jumping to a specific node: In some scenarios the view was wrong*
    - *Fix bug in error panel date format*
    - *Fix bug downloading a PDF when the server does not have Internet access*
    - *Fix bug in form elements with HorizontalPanel or VerticalPanel in root definition*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

v7.1.18

- *Release: 2020-11-03*
- *Added comment filed in automation*
- *Fixed problem with attachment in asian languages*
- *Fixed issue in PDF Tools*
- *Improvements in record lock*
- *Improvements and fixes in KCenter:*
    - *Improvement in toolbar buttons and contextual menu options: disabled options are now clearly marked in grey*
    - *Improvement in main menu: now you can quickly change the current tenant*
    - *Added "force lock" and "force unlock" for documents and records*
    - *Added PDF Tools: You can extract, rotate, split and delete pages from any PDF*
    - *Added massive preview feature.*
    - *Fixed bug in notes: the edit button should not be shown if the user is not allowed to edit the note*
    - *Fixed bug in the path shown in email attachments*
    - *Fixed bug with keywords and special characters like & and < or >*
    - *Fixed bug with document stamp: the rotation shown in the UI had wrong direction*
    - *Fixed bug with massive actions in search view: when you change between "Default view" and "Compact view" the selected checkbox still remains selected*
    - *Fixed bug in wofklow form elements: dynamic value mapping now works*
    - *Fixed bug in select user popup (usually used when changing node security)*
    - *Fixed bug in filebrowser when using Google Chrome: selected row was shown with a black border*

v7.1.17

- *Release: 2020-10-29*
- *Fixed error when editing Automation rules*
- *Fixed error when using personal images on stamp*
- *Fixed compatibility issue with doc-signer*
- *Added userName to sendMail Automation*
- *Improvements and fixes in KCenter:*
    - *Added refresh option in e-mail browser*
    - *Added "Text extracted " tab on e-mails*
    - *Added advanced filtering to choose user and roles*
    - *Added massive contextual menu in search view*
    - *Added clone of security and metadata in copy action*
    - *Added "Merge to PDF" in massive actions*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Fixed bug in workflow pooled tasks*
- *Fixed bug in choose destination path when copy or move: It's not allowed to copy or move to trash*
- *Fixed bug in choose destination path when copy or move: It's not allowed to copy or move to categories when the node to copy or move is not a category*
- *Fixed bug in form validation plugin*
- *Fixed bug in search view which was not selecting the current node*
- *Fixed bug in search view which was not showing a date range in date metadata fields*

v7.1.16

- *Release: 2020-09-24*
- *Fixed problem with mail thumbnail generation*
- *Fixed issue when importing emails not RCF-2183 compliant*
- *New electronic signature verification module*
- *New concurrent mail extraction process*
- *Improvements and fixes in KCenter:*
  - *Browser view, "Merge PDF" option added (available when you select several documents)*
  - *Browser view: Massive "Add metadata group" option added*
  - *Browser view: Massive "Change security" option added*
  - *Browser view: "Save as PDF" option added*
  - *Browser view: Empty trash action moved to left vertical menu*
  - *Browser view: New icon for subscribed nodes*
  - *Email preview: New contextual menu to download and copy attachments*
  - *New autoLogin screen which provides detailed information about the login process*
  - *New Lucene advanced query (disabled by default in user profiles)*
  - *Removed ugly scroll when downloading a document*
  - *Fixed paths from e-mails and attachments*
  - *The user who created the relation can remove it*

v7.1.15

- *Release: 2020-08-31*
- *Fixed database error in background pending task process*
- *Fixed bug in antivirus background execution task*
- *Fixs and improvements in massive import API*
- *Check-in & check-out implemented in CMIS*
- *Improved loggin in repository import*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Improved XLSX text extraction*
- *Improvements and fixes in KCenter:*
    - *New HTML editor*
    - *New contextual menu in browser view*
    - *Digital Signature Viewer extension is now supported*
    - *Activity log extension is now supported*
    - *Extra tab folder, record and e-mail extensions are supported*
    - *Minor usability improvements: toolbar is always visible in browser*
    - *Fixed error when trying to download files and the server does not have Internet connection*
    - *Fixed preview annotation issue*
    - *Fixed parent-children metadata selected issue*
    - *Fixed default metadata values set in the definition are selected on creation*
    - *Fixed workflow in the wizard*
    - *Administrators can cancel any checked-out document*
    - *Administrators can unlock any locked document*
    - *Default view in profile now it takes effect in the user interface*
    - *Fixed pooled workflows error in the dashboard view*
    - *Readonly attribute in the metadata now works*
    - *Properties panel now shows the user who edits the document*

v7.1.14

- *Release: 2020-06-17*
- *Improved HTML text extraction*
- *Improved document version comparison*
- *Improved REST API error handling (need new SDK v3.8)*
- *Fixed problem with quick search in KCenter*
- *Fixed thumbnail generation error with SQL files*
- *Fixed random issues with massive import API*
- *Experimental thumbnail performance improvement*
- *New feature: assigned tasks are now shown in KCenter dashboard*
- *New feature: document and metadata history preview in KCenter*
- *New feature: electronic mail digital signature*
- *New feature: Windows Live Edit client*
- *New feature: datastore sharding*

v7.1.13

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Release: 2020-05-14*
- *Fixed error when purging trashes from administration*
- *Fixed NPE when sending mail with attachments*
- *Fixed page break TinyMCE plugin*
- *Implemented REST mail download*
- *Added button to resolve XREFs in AutoCAD files*
- *New KCenter look and improved usability*
- *Removed database repository backend*
- *Performance improvements*
- *Improved lock feature*
- *Improvements and fixes in the KCenter user interface:*
    - *Changed the colour palette*
    - *Improved UI buttons and panels location*
    - *Direct drag & drop from file explorer*
    - *Fixed bug searching with characters "&" and "+"*

v7.1.12

- *Release: 2020-04-20*
- *Fixed issue with File Plan*
- *Fixed problem when importing mails*
- *Fixed problem with special characters in node name*
- *Mail raw is always stored to preserve traceability*
- *Pagination enabled by default in classic UI*
- *Improved mail import performance*
- *Make subscription notification optional*

v7.1.11

- *Release: 2020-03-23*
- *Improved mail download*
- *Improved mail previsualization*
- *Node notes can be sort by date*
- *Workflow diagram is shown again*
- *Several perfomance improvements*
- *New metadata browser in KCenter*
- *Several changes and improvements in KCenter*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

v7.1.10

- *Release: 2020-01-27*
- *Implemented FTP support*
- *Implemented Markdown previsualization*
- *Fixed problem when importing mails without body*
- *Fixed issue in REST API when username has a dot*
- *Fixed security issue with authentication token*
- *Fixed issue with workflow task notifications*
- *Fixed error with download mail attachments in KCenter*
- *Fixed issue with reports in KCenter*
- *Unzip implemented in KCenter*
- *Document stamp implemented in KCenter*

v7.1.9

- *Release: 2019-12-19*
- *New cluster mode implemented*
- *Strict search implemented for metadata*
- *Workflow entities managed from administration*
- *Security improvements in administration*
- *Support for workflows added in the new UI*
- *Support for CADViewer preview in the new UI*
- *Many other improvements and fixes in the new UI*

v7.1.8

- *Release: 2019-11-26*
- *Several CADViewer integration improvements*
- *Increaed Hibernate version to 5.1.17.Final*
- *OCR Template images stored in filesystem*
- *Automatic language detection removed*
- *New methods for massive imports*
- *Metadata browsing is now fixed*
- *Improvements in auth token*
- *Fix NPE in OCR Capture*
- *New KCenter included*

v7.1.7

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Release: 2019-10-23*
- *Improvements in reports*
- *Performance Improvements*
- *Improved log configuration*
- *Several metadata error fixed*
- *Purge obsolete or orphan information*
- *Predictables indexes and foreign keys names*
- *Files openkm.xml and ehcache.xml are no optionals*
- *Records wasn't able to be copied or moved massively*

v7.1.6

- *Release: 2019-09-16*
- *Added PDF Toolbox*
- *Upgrade to CADViewer 4.1*
- *Fixed problem with repository import*
- *Improved SQL Server compatibility*
- *Improved Oracle compatibility*
- *Improved Diff PDF feature*
- *Added more complete CMIS support*
- *Fixed document history comparison*
- *Fixed issue with background zip download*
- *Added basic authentication for REST methods*
- *Implemented download toke for previsualization*
- *Lots of improvements and fixes in medatada feature*
- *Filter by metadata was not working when the text in uppercase*
- *Added automation event when importing mail with attachments*
- *TabWorkspace export UUIDs of selected nodes as a parameters*

v7.1.5

- *Release: 2019-06-19*
- *First public release.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Updater utility

OpenKM "Updater utility" is a Java Administrator tool that helps with performing upgrades automatically. You can download it from http://download.openkm.com/okm/OKMUpdater.jar.

*This utility needs access to **update.openkm.com** and **download.openkm.com** domains.*

Updater utility requires Java 8 >= 8u101

Command line parameters:

Parameter	*Description*
-h	*Show help information.*
-d	*Update development version (should only used under OpenKM technicals supervision).*
-f	*Force the update.*
-l	*Remove the Tomcat logs.*
-m	*Apply actions manually.*
-n	*Monochrome mode.*
-o	*Disable post-stage actions.*
-r	*Disable pre-stage actions.*
-u	*Override detected Tomcat URL.*
-v	*Show version.*

Upgrade stages

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Check if it's located at $TOMCAT_HOME folder, otherwise it will raise an error message.*
- *Ask for some information from the administrator.*
    - *OpenKM URL (usually someone accessible from server like http://localhost:8080/openkm).*
    - *The application is configured as service [Y/N].*
    - *The OpenKM valid ROLE_ADMIN user (okmAdmin).*
    - *The administrator user password.*
- *Check the service configuration, the user connection and the OpenKM version.*
- *If the utility is successful on these checks, it will require confirmation by the user to proceed with upgrade [Y/N].*
- *Download the new openkm.war file (current version + 1).*
- *When finished downloading, it will execute pre-upgrade actions.*
- *Stop OpenKM.*
- *Replace the old OpenKM.war file by the new downloaded one.*
- *Modify the parameter **hibernate.hbm2ddl** to **update** into the file $TOMCAT_HOME/openkm.properties.*
- *Start OpenKM again.*
- *Check if OpenKM is already available and then execute post-upgrade actions.*

Manual mode

*If you invoke the OKMUpdater utility with the **-m** parameter, **none of the actions will be executed, so you need to executen them manually**. Depending on the action type you need to use different utilities located at **Administration > Utility**.*

It is very important to execute the actions, and execute them in the right order, otherwise it may cause information loss.

Types of actions:

- ***SQL**: Go to **Administration > Utilities > Database query** (Choose SQL from select)*
- ***HQL**: Go to **Administration > Utilities > Database query** (Choose HQL from select)*
- ***Script**: Go to **Administration > Utilities > Scripting**.*

Keep on mind that some actions should be executed before the upgrade and others after the upgrade. This is the upgrade process in short:

- ***Execute PRE actions***
- *Stop OpenKM*
- *Replace openkm.war*
- *Start OpenKM*
- ***Execute POST actions***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

It is important to know which actions are executed in PRE and in POST.

Sometimes maybe there is no PRE or POST actions

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Updater utility english video

Your browser does not support the video tag.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Updater utility spanish video

Your browser does not support the video tag.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Tomcat update

The upgrade procedure described below is valid for Tomcat versions:

- *8.5.57*

By default, OpenKM is installed with a specific Tomcat version. But every few weeks a new version is released and maybe you want your installation up to date. In this case, should open a ticket in the OpenKM support website to ensure this new version has been packed by the OpenKM Team. Once you have verified and downloaded the new version is available you have to follow the steps described in this section of the documentation.

Before updating Tomcat we recommend to upgrade to the latest OpenKM version and ensure you have a recent working backup.

Although you can use a pristine Tomcat we always recommend using those provided by OpenKM team because have additional files and default configurations which work best with OpenKM.

Linux

Preliminaries

- *Check in the **openkm.properties** file if you have a special repository path location (more information at Change repository home).*

Tomcat replacement

- ***Stop the current Tomcat***
- ***Unzip the new Tomcat***
- ***Copy** these **files and folders from the old Tomcat to the new** one:*

Remember when copying and moving data to check the security applied in the destination. Sometimes when you move data the security applied might change and what cause issues in the OpenKM startup process.

You can set them recursively:

```
$ sudo chown openkm:openkm tomcat-8.5.47 -R
```

Name	Type	Comments
tomcat-XXX/bin/setenv.sh	File	Optional. Only required if you have modified this file.

tomcat-XXX/openkm.properties	*File*	*OpenKM configuration file.* *If there's a mistake copying this file, might damage the repository. Please check the file has been correctly copied into the destination.*
tomcat-XXX/openkm.xml	*File*	*OpenKM security configuration file.* *This file might not exists. You can find this file only in some special installation.*
tomcat-XXX/logback.xml	*File*	*OpenKM log configuration file. Might not exists.*
tomcat-XXX/repository	*Folder*	*The folder where documents are stored.* *If there's a mistake copying this folder, might damage the repository. Please check the file has been correctly copied into the destination.*
tomcat-XXX/plugins	*Folder*	*Directory with OpenKM plugins.* *Might be empty. It only contains files if the OpenKM instance has customizations.*
tomcat-XXX/webapps	*Folder*	*OpenKM deployed application.*
tomcat-XXX/jbpm-console.properties	*File*	*jBPM engine configuration file.* *This file might not exists. This file exists only if jBPM engine has been installed.*

- ***Update your Tomcat service configuration**:*
  - *Change the **Tomcat path** at **/etc/init.d/openkm***
- ***Start OpenKM***
- *Go to Administration > Configuration parameters*
- ***Filter** in the top right corner **by the word "tomcat"***
- ***Replace the old Tomcat path** used in configuration parameters **by the new path**.*
- ***Restart OpenKM service.***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***Do a quick test.***

Suggested quick test:

- *Browse the repository.*
- *Download an existing file.*
- *Upload a new file a try to download.*

Rememeber to update the backup script because the location of the Tomcat directory would change. In case of Linux you can check it by doing (as root):

```
$ crontab -l
```

This way, you can see the location of the backup script.

Preserving the old Tomcat is a good idea, but keep on mind that the backup script usually backups all the /home/openkm directory so move it to another place.

If you want a more recent branch you can try Tomcat 9.0, but keep on mind that this new branch has not been fully tested and is not supported by OpenKM team.

Windows

Preliminaries

- *From services check the user who's executing OpenKM. Usually, the user should be "Local services", but sometimes it is used a standard user. Remember when registering the OpenKM service again to choose the right user.*
- *Check in the openkm.properties file if you have a special repository path location (more information at Change repository home).*

Tomcat replacement

- ***Stop OpenKM from services***.
- ***Delete OpenKM from services.***

Two options to delete OpenKM from services:

First option:

- *Open a terminal with Administrator user.*
- *Go to tomcat-XXX/bin folder.*
- *Execute the next command:*

```
$:> service.bat remove
```

Second option:

- *Open a terminal with Administrator user.*
- *Execute the next command:*

```
$:> sc delete OpenKM
```

- ***Unzip de new Tomcat.***
- ***Copy** these **files and folders from the old Tomcat to the new** one:*

Remember when copying and moving data to check the security applied in the destination. Sometimes when you move data the security applied might change and what cause issues in the OpenKM startup process.

Name	*Type*	*Comments*
tomcat-XXX\bin\setenv.bat	*File*	*Optional. Only required if you have modified this file.*
tomcat-XXX\openkm.properties	*File*	*OpenKM configuration file.* *If there's a mistake copying this file, might damage the repository. Please check the file has been correctly copied into the destination.*
tomcat-XXX\openkm.xml	*File*	*OpenKM security configuration file.* *This file might not exists. You can find this file only in some special installation.*
tomcat-XXX\logback.xml	*File*	*OpenKM log configuration file. Might not exists.*
tomcat-XXX\repository	*Folder*	*The folder where documents are stored.* *If there's a mistake copying this folder, might damage the repository. Please check the file has been correctly copied into the destination.*
tomcat-XXX\plugins	*Folder*	*Directory with OpenKM plugins.* *Might be empty. It only contains files if the OpenKM instance has*

		customizations.
tomcat-XXX\extras	*Folder*	*Third-party applications used by OpenKM*
tomcat-XXX\webapps	*Folder*	*OpenKM deployed application.*
tomcat-XXX\jbpm-console.properties	*File*	*jBPM engine configuration file.* *This file might not exists. This file exisit only if jBPM engine has been installed.*

- ***Configure OpenKM as service***

- *Open a terminal with Administrator user.*
- *Go to tomcat-XXX/bin folder.*
- *Execute the next command:*

```
$:> service.bat install
```

- *Execute the next command:*

```
$:> openkm //US//OpenKM ++JvmOptions="-XX:+UseG1GC"
$:> openkm //US//OpenKM ++JvmOptions="-XX:+UseStringDeduplication"
$:> openkm //US//OpenKM ++JvmOptions="-XX:+HeapDumpOnOutOfMemoryError"
$:> openkm //US//OpenKM ++JvmOptions="-XX:HeapDumpPath=$CATALINA_HOME"
$:> openkm //US//OpenKM ++JvmOptions="-Djava.net.preferIPv4Stack=true"
$:> openkm //US//OpenKM ++JvmOptions="-Dmail.mime.ignoreunknownencoding=true"
$:> openkm //US//OpenKM ++JvmOptions="-Dfile.encoding=utf-8"
$:> openkm //US//OpenKM ++JvmOptions="-Dsun.jnu.encoding=UTF-8"
$:> openkm //US//OpenKM ++JvmOptions="-Djava.awt.headless=true"
$:> openkm //US//OpenKM ++JvmOptions="-
Dsun.java2d.cmm=sun.java2d.cmm.kcms.KcmsServiceProvider"
```

- ***Start OpenKM service***
- *Go to Administration >* *Configuration parameters*
- ***Filter*** *in the top right corner* ***by the word "tomcat"***.
- ***Replace the old Tomcat path*** *used in configuration parameters* ***by the new path***.
- ***Restart OpenKM service.***
- ***Do a quick test.***

Suggested quick test:

- *Browse the repository.*
- *Download an existing file.*
- *Upload a new file a try to download.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Manual update

In the whole public documentation, you will not find any explanation about how doing a manual upgrade.

The whole migration section and the OpenKM Updater tool has been created with the purpose of helping users in the upgrading OpenKM process. Unfortunately the automatic upgrade tools

Unfortunately, the automatic upgrade tools are not compliance with all the companies policies and in some cases is mandatory do a manual upgrade.

We have decided not explaining here because although almost migration steps between OpenKM versions are easy, someone's needs you pay attention to them. For better service and to keep safe your system we think working together on it is the best scenario.

If this is your case, please create a ticket at our OpenKM customer support website and we will help you on it.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Cluster update

Since OpenKM v7.0.8 there is a new and much better cluster architecture. To take advantage of these clustering improvements you should upgrade your cluster configuration.

Master node

We have to modify the openkm.properties configuration file, so you have to stop Tomcat:

```
$ sudo /etc/init.d/tomcat stop
```

Once Tomcat has been shutted down, remove the Lucene indexes:

```
$ rm -rf $TOMCAT_HOME/repository/index
```

Edit the openkm.properties file and modify the cluster related section:

```
# Cluster - Master
cluster.backend=jms
cluster.node=master
```

*Now edit the **$TOMCAT_HOME/conf/context.xml** Tomcat resources configuration:*

```
<!-- Cluster - Master & Slave -->
<ResourceLink global="jms/ConnectionFactory" name="jms/ConnectionFactory" type="javax.jms.ConnectionFactory"/>
```

*And the **$TOMCAT_HOME/conf/server.xml** resources configuration file:*

```
<!-- Cluster -->
<Resource auth="Container" name="jms/ConnectionFactory" type="org.apache.activemq.ActiveMQConnectionFactory"
    description="JMS Connection Factory" factory="org.apache.activemq.jndi.JNDIReferenceFactory"
    brokerURL="tcp://master:61616" brokerName="HibernateSearchBroker"/>
```

Start Tomcat again:

```
$ sudo /etc/init.d/tomcat start
```

Login into OpenKM and go to Administration > Utilities > Rebuild indexes in order to generate the Lucene indexes again.

Slave nodes

We need to stop Tomcat because we have to modify the openkm.properties configuration file:

```
$ sudo /etc/init.d/tomcat stop
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

When Tomcat is stopped remove the Lucene indexes:

```
$ rm -rf $TOMCAT_HOME/repository/index
```

Edit the openkm.properties file and modify this section:

```
# Cluster - Slave
cluster.backend=jms
cluster.node=slave
```

*Edit the **$TOMCAT_HOME/conf/context.xml** Tomcat resources configuration:*

```
<!-- Cluster - Master & Slave -->
<ResourceLink global="jms/ConnectionFactory" name="jms/ConnectionFactory" type="javax.jms.ConnectionFactory"/>
```

*Also edit the **$TOMCAT_HOME/conf/server.xml** resources configuration file:*

```
<!-- Cluster -->
<Resource auth="Container" name="jms/ConnectionFactory" type="org.apache.activemq.ActiveMQConnectionFactory"
    description="JMS Connection Factory" factory="org.apache.activemq.jndi.JNDIReferenceFactory"
    brokerURL="tcp://localhost:61616" brokerName="HibernateSearchBroker"/>
```

Start Tomcat again:

```
$ sudo /etc/init.d/tomcat start
```

Login into OpenKM and go to Administration > Utilities > Rebuild indexes in order to generate the Lucene indexes again.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Migrating from 7.1.3 to 7.1.4

Migration

Before the update

> ***Review your metadata fields definition name*** *in version 7.1.4 some names will no be longer supported.*
>
> ***Only a single character "." is allowed in the field name*** *.*

From version 7.1.4 fields name should be like sample below:

- *The group is named okg:contact_test.*
- *The field is named okp:contact_test.first_name (only one character "." separator between the group name and field name).*

```
<property-group label="Contact test" name="okg:contact_test">
      <input label="Firt Name" name="okp:contact_test.first_name"/>
</property-group>
```

Sample of wrong metadata definition below:

- *The field named* ***okp:contact_test.first.name*** *should be renamed as* ***okp:contact_test.first_name*** *with a single "." separator.*

```
<property-group label="Contact test" name="okg:contact_test">
      <input label="Firt Name" name="okp:contact_test.first.name"/>
</property-group>
```

If you have this issue in your current metatada definition follow the next steps:

- *Create a list of current metadata value an for what will be changed*

```
okp:contact_test.first.name -> okp:contact_test.first_name
```

- *Create SQL queries based in your metadata field changes:*

```
UPDATE OKM_NODE_PROPERTY SET NPG_NAME='okp:contact_test.first_name' WHERE NPG_NAME =
UPDATE OKM_NODE_PROPERTY_VERSION SET NPV_NAME='okp:contact_test.first_name' WHERE NPV
```

- *Execute the SQL queries from Administration > Tools > Database query*

Update process

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Use the OKMUpdate.jar tool to upgrade version 7.1.3 to version 7.1.4*

After the update

- *Because in version 7.1.4 has been improved the metadata history, might be necessary manual intervention in this upgrade.*

Because there is an issue with database select with BigInteger when using cursor (large queries), the field NPV_VERSION might be converted to VARCHAR

Step 1 - Database changes

MySQL

```
ALTER TABLE OKM_NODE_PROPERTY_VERSION DROP FOREIGN KEY FKnm8i9xa5uklqh8yyysognwe1h;
ALTER TABLE OKM_NODE_PROPERTY_VERSION MODIFY NPV_VERSION varchar(64);
ALTER TABLE OKM_NODE_PROPERTY_VERSION DROP INDEX IDX_NPVERSION_VERSION;
```

Step 2 - Execute the script

You can follow what the script is doing in the file $TOMCAT_HOME/logs/MetadataHistoryDataMigration_yyyyMMdd.log.

```
import org.springframework.web.context.*;
import org.springframework.web.context.*;
import com.openkm.module.db.stuff.*;
import com.openkm.util.*;
import org.hibernate.*;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.core.Config;
import com.openkm.api.*;
import com.openkm.bean.*;
import java.util.*;

String LOG_FILE = "MetadataHistoryDataMigration";
WebApplicationContext cc = (WebApplicationContext) ContextWrapper.getContext();
OKMPropertyGroup okmPropertyGroup = cc.getBean(OKMPropertyGroup.class);
PropertyGroupStorage propertyGroupStorage = cc.getBean(PropertyGroupStorage.class);
SessionFactory sessionFactory = cc.getBean(SessionFactory.class);
StatelessSession session = sessionFactory.openStatelessSession();
StatelessSession session2 = sessionFactory.openStatelessSession();
StatelessSession session3 = sessionFactory.openStatelessSession();

FileLogger.info(LOG_FILE, ">>>STARTING METADATA HISTORY DATA MIGRATION");
String systemToken = DbSessionManager.getInstance().getSystemToken(Config.DEFAULT_TEN.
List groups = okmPropertyGroup.getAllGroups(systemToken);
for (PropertyGroup group : groups) {
    String grpName = group.getName();
 FileLogger.info(LOG_FILE, ">>>Processing group named: " + grpName);
 String sql = "select DISTINCT(npv.NPV_VERSION) from OKM_NODE_PROPERTY_VERSION npv WH.
 Query query = session.createSQLQuery(sql);
 query.setReadOnly(true);
 query.setFetchSize(10);

 ScrollableResults results = query.scroll(ScrollMode.FORWARD_ONLY);
```

```
 while (results.next()) {
  String npvVersion = results.get(0);
  FileLogger.info(LOG_FILE, ">>>Version to process: " + npvVersion);
  Query queryNodeVersion = session2.createSQLQuery("SELECT NV_PARENT FROM OKM_NODE_VE
  List nodeVersion = queryNodeVersion.list();
  if (nodeVersion.size() == 1) {
   String uuid = nodeVersion.get(0);
            FileLogger.info(LOG_FILE, ">>>Processing history version with NV_ID: " +
   print(uuid+"<br/>");
   Map properties = new HashMap();
   Query queryNodeVersionProperties = session3.createSQLQuery("SELECT NPV_NAME, NPV_V
   List nodeVersionProperties = queryNodeVersionProperties.list();
   for (Object[] propertiesValues : nodeVersionProperties) {
    String propertyName = propertiesValues[0];
    String propertyValue = propertiesValues[1];
    properties.put(propertyName, propertyValue);
   }
            String valuesToString = properties.toString().replaceAll("\\{","[[").repl
   FileLogger.info(LOG_FILE, ">>>PropertyValues for NV_ID: " + npvVersion + " with va
   print(properties);
   propertyGroupStorage.addPropertyGroupVersion(uuid, Long.parseLong(npvVersion), grp
  } else {
            FileLogger.error(LOG_FILE, ">>>Version NV_ID: " + npvVersion + " not foun
  }
 }

    results.close();
}
session.close();
session2.close();
session3.close();
FileLogger.info(LOG_FILE, "<<<ENDED METADATA HISTORY DATA MIGRATION");
```

Step 3 - Final database changes

MySQL

```
DROP TABLE OKM_NODE_PROPERTY_VERSION;
```

Step 3 - Reindex the whole repository

- *Go to Administration > Tools > Rebuild indexes*
- *Check the Lucene indexes option.*
- *Click on "Execute" button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Changes from version 6.4

Changes

- *The **jBPM workflow engine** is not embedded into OpenKM. Follow steps described at Configuring jPBM.*
- *A lot of improvements and changes in the **REST API**.*
- ***Language detection** is **disabled by default**.*
- *Parameters in **OpenKM.cfg** with value "**on**", have value "**true**" in **openkm.properties***
- *Because we consider the Database storage feature is conceptually a malicious feature which always provides poor performance, definitively has been removed. That's why the configuration parameter "**repository.datastore.backend**" has been removed.*
- *The password of the user stored in the database in **md5** encryption has been changed to **bcrypt** encryption.*
- ***Security access manager** works with a plugin architecture. More information at Creating your own security access manager and Security configuration.*
- *Deprecated parameter named "**hibernate.search.index.home**" take a look at Change repository home.*
- *Deprecated configuration parameter named "**ask.drag.and.drop.updates**". From version 7.1.21 it is replaced by a configuration parameter in the profile.*

Deprecated features from version 6.4

- ***SOAP** services have been removed.*
- ***Database metadata values** support has been removed (no longer support for tables **OKM_DB_METADATA_TYPE**, **OKM_DB_METADATA_VALUE** and **OKM_DB_METADATA_SEQUENCE)** .*
- ***OMR** has been removed from OpenKM core now it is an external module.*
- ***Joomla integration** has been removed.*
- ***Wordpress integration** has been removed.*
- ***Terminal Shell plugin** has been removed.*
- ***Google Drive integration** has been removed.*
- ***Macros extension** has been removed*
- ***Stapling extension** has been removed in favour of relations feature.*
- ***Contact** extension has been removed from version 7.1.21*
- ***Wiki** extension has been removed from version 7.1.21*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Development

OpenKM professional portable development environment

The OpenKM Professional Portable development environment comes with all in one configured tools.

> *We encourage using our OpenKM portable development environment rather configure your own Eclipse, specially for Java beginners.*
>
> *We've tried to provide a development platform as easy as possible to get it running with few steps.*
>
> *More information about OpenKM professional portable development environment at Download center.*

Tools:

- *Eclipse*
- *OpenJDK 8*
- *MariaDB*
- *DBeaver*
- *iReport*
- *Tesseract*
- *LibreOffice*
- *Geany*
- *jBPM 3.3.1.GA*
- *Firefox*

Code:

- *Automation samples.*
- *Crontab samples.*
- *Form samples.*
- *OCR samples.*
- *Report samples.*
- *Workflow course.*

Suggested minimum hardware configuration:

- *2 CPU's*
- *4GB RAM*
- *64 bits architecture*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Javadoc

For each OpenKM version, it is created a complete Javadoc available at http://docs.openkm.com/apidoc/.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Personal development environment

OpenKM personal development environment is distributed as a virtual machine image (OVA).

The personal development environment comes with everything you need to start quickly working:

- *OS Linux Mint distro.*
- *OpenKM application installed*
- *MariaDB database*
- *Eclipse*
  - *Code samples*
  - *Workflow samples*
- *IReport*
  - *SQL samples*
  - *Scripting samples*

Requirements

The virtual machine uses 2 core and 4GB of RAM, the host machine should have at least 4 core and 8GB of RAM.

We suggest having almost applications stopped while executing the virtual machine.

We suggest installing on a desktop computer either laptop computer. Although can working in laptop but with good hardware.

We suggest SSD disk to getting an extra performance in the virtual machine.

In some users, we observe executing the virtual machine in a secondary disk provides extra performance, seems some motherboards do not like the OS and Virtualization in the same hard disk.

How to get the personal development environment

Can download the virtual machine file from OpenKM download center.

- *Go to https://download.openkm.com*
- *Click on the button "OpenKM Professional Download Center".*
- *Login with your customer credentials.*
- *In the top menu choose tools.*
- *In the tools, ensure you have chosen OpenKM version 7.1.x*
- *Click in the Open KM-PorDev.ova file*

Download link is sent by e-mail.

The link can be used only one time.

You have 24h from the e-mail have been sent to use the link, passed this time the link is expired.

How to install

The OpenKM personal development environment can be installed in almost virtual machine software like Virtual Box, VMWare, Hyper-V.

Because OVA (Open Virtual Appliance) seems is not 100% compatible between virtualization software at least users of VMWare usually should make few modifications in the file or extract the mkv to import the virtual machine.

If you are looking for a quick deploy we encourage using Virtual Box.

If you decide for Virtual Box, follow the steps described in the next video *https://www.youtube.com/watch?v=9ugBYf5ggu8&feature=emb_logo*

License activation

Personal development environment must have Internet connection to register the license.

Finally, the license of your personal development environment must be activated, for it follow the next steps:

1- Open a terminal (with user openkm)

2- Execute the command

```
$> pdl-reset
```

3- Start OpenKM service

```
$> sudo service openkm start
```

4- Open an issue in the OpenKM customer support website. Share the application ID available in the home of the administration.

How to upgrade the development environment

The process for upgrading the development environment is exactly the same of upgrading standard OpenKM professional edition. More information at Updater utility.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring IDE from a clean Eclipse

We recommend using our OpenKM Portable Development IDE rather than configuring your own.

You can also use IntelliJ IDEA or other development Java tools like NetBeans.

Requirements

Software:

- *Eclipse*
- *OpenJDK 1.8*

Your eclipse must have support for:

- *Maven*
- *Subversion (recommended)*

Hardware:

Suggested minimum hardware configuration:

- *2 CPU's*
- *4GB of ram*
- *64 bits architecture (although you can get running on 32 we do not suggest this architecture).*

Download eclipse

*We suggest downloading the latest stable version of Eclipse. Eclipse comes with several versions, you should download the "**Eclipse IDE for Java EE Developers**" for your computer architecture (32 or 64 bits).*

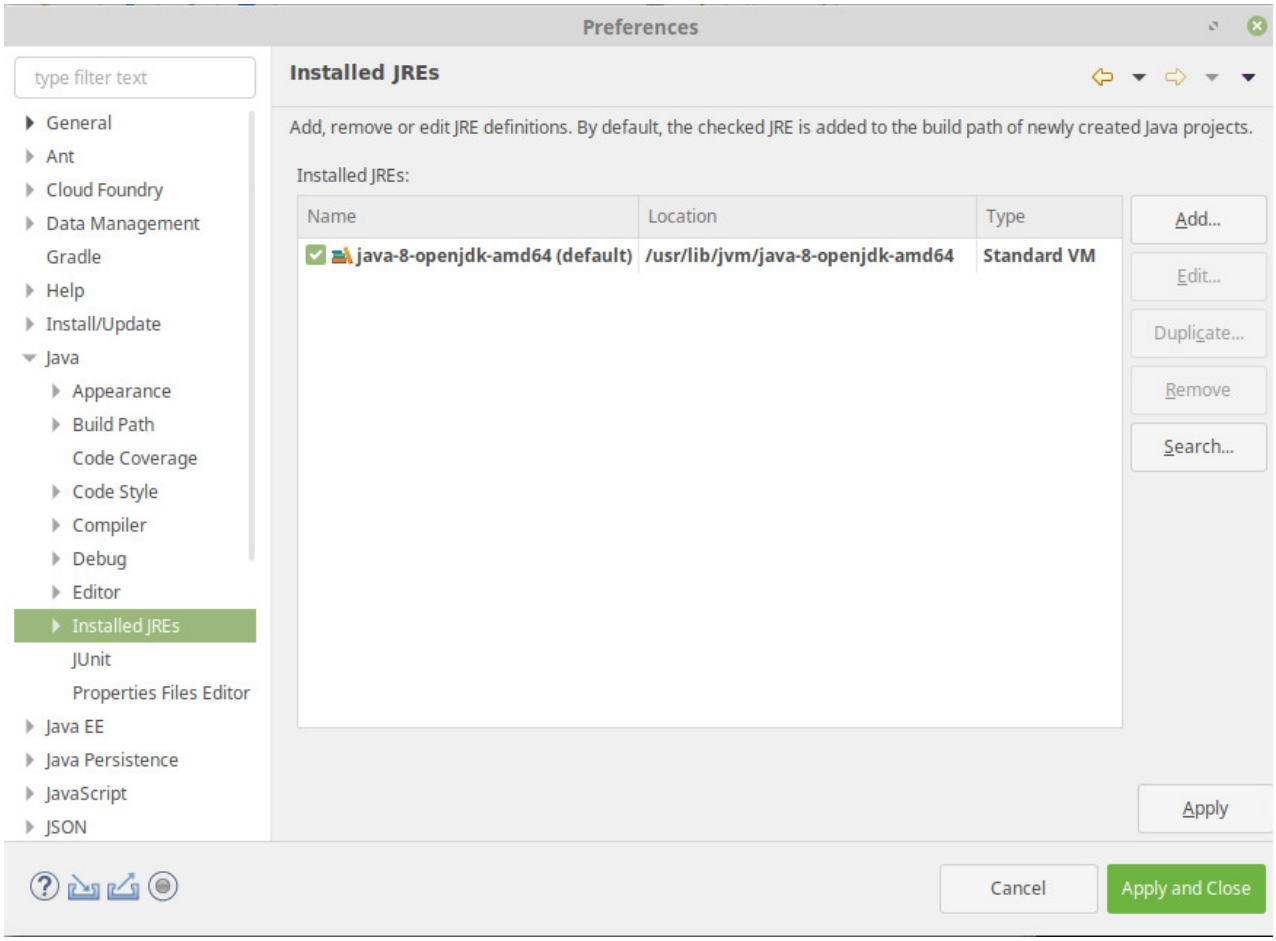
More information at https://eclipse.org/downloads/.

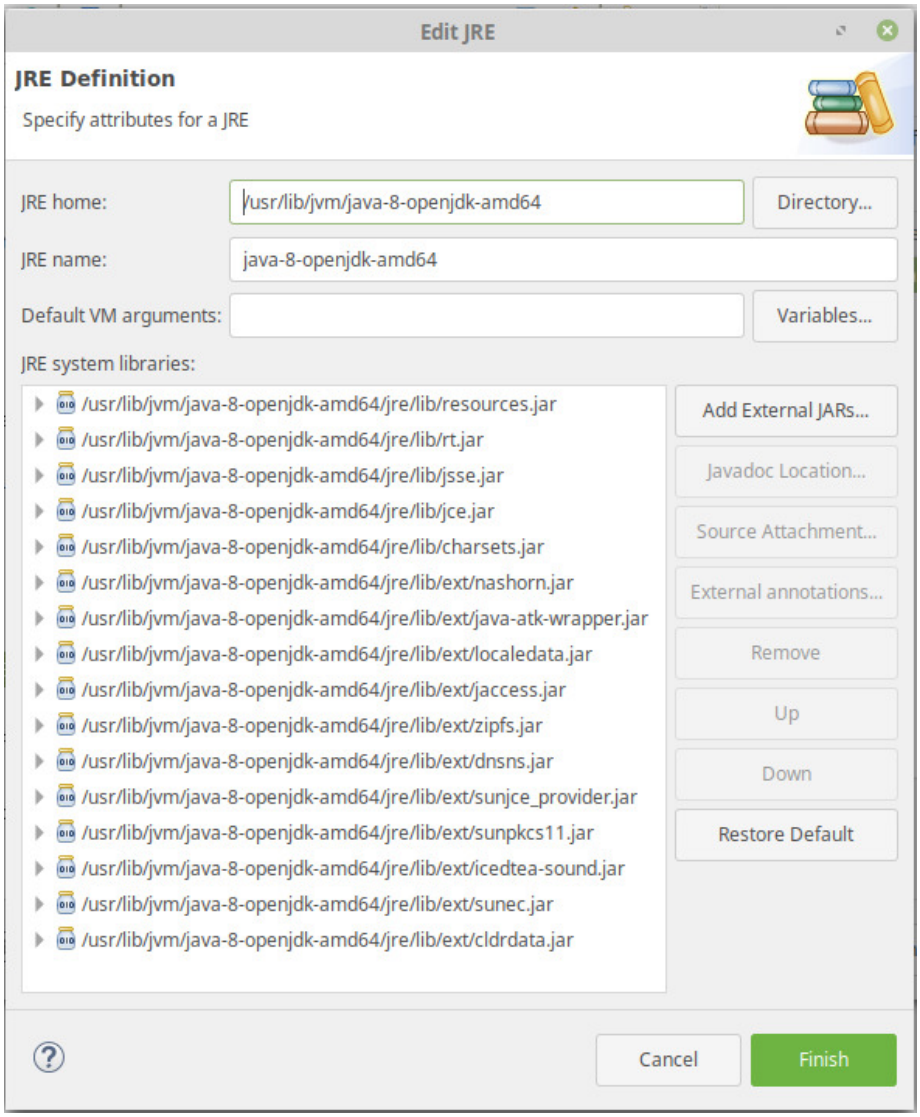
Configure Java

You must have Java installed in your server otherwise go to http://www.oracle.com/technetwork/java/index.html click at Java SE and download the latest OpenJDK 1.8 version accorder your OS and hardware architecture. Execute the Java installer and follow the installation wizard.

*Go to **Window** / **Preferences** / **Java** / **Installed JRE**. You must have OpenJDK 1.8 defined here:*

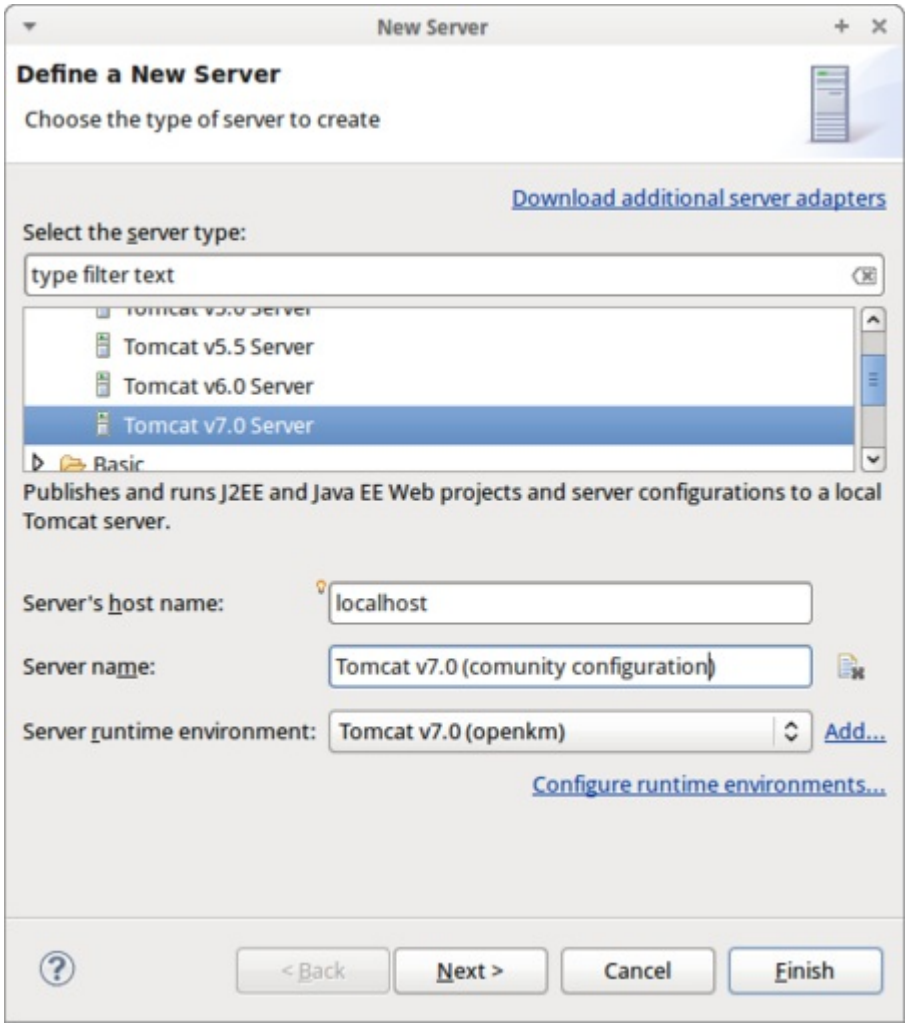
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


Preferences
type filter text
General
Ant
Cloud Foundry
Data Management
Gradle
Help
Install/Update
Java
Appearance
Build Path
Code Coverage
Code Style
Compiler
Debug
Editor
Installed JREs
JUnit
Properties Files Editor
Java EE
Java Persistence
JavaScript
JSON
Installed JREs
Add, remove or edit JRE definitions. By default, the checked JRE is added to the build path of newly created Java projects.
Installed JREs:
Name
Location
Type
java-8-openjdk-amd64 (default)
/usr/lib/jvm/java-8-openjdk-amd64
Standard VM
Add...
Edit...
Duplicate...
Remove
Search...
Apply
Cancel
Apply and Close



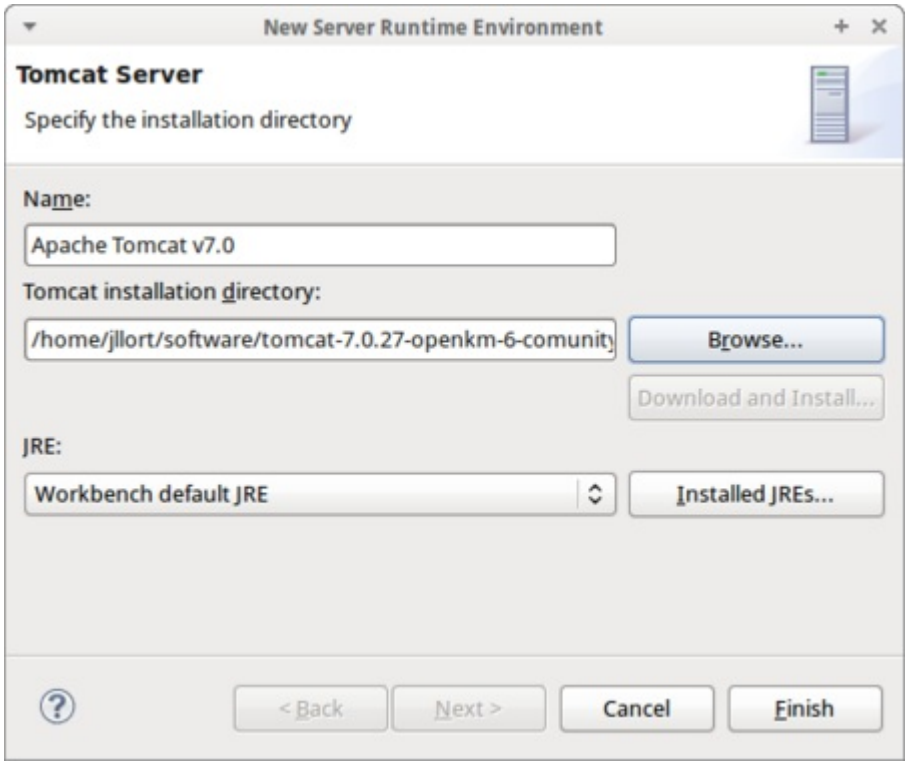
Configure tomcat server

*Go to "**servers**" tab. Click on right click mouse button and from contextual menu choose "**Add / server**". From the list choose "**Tomcat 7 server**":*



*Configure a new **Run time server environment**:*

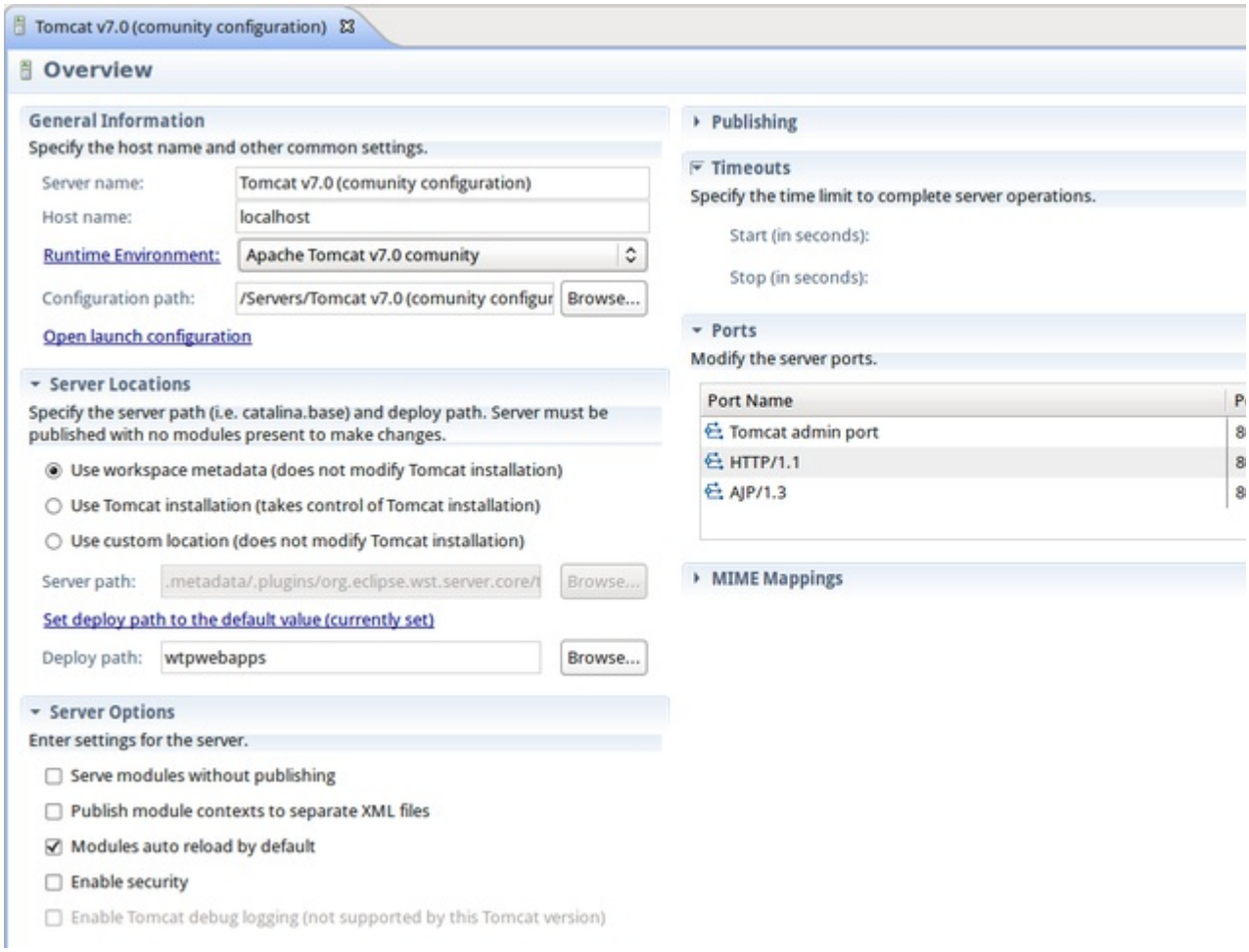
> *Choose the "Workbench default JRE" when your default JRE is "OpenJDK 1.8" or choose "OpenJDK 1.8" from JRE list of installed JREs.*



*Check box **Use Tomcat installation** (takes control of tomcat installation).*

*Increment start and stop **timeouts** (for example to 180 and 120 seconds respectively).*

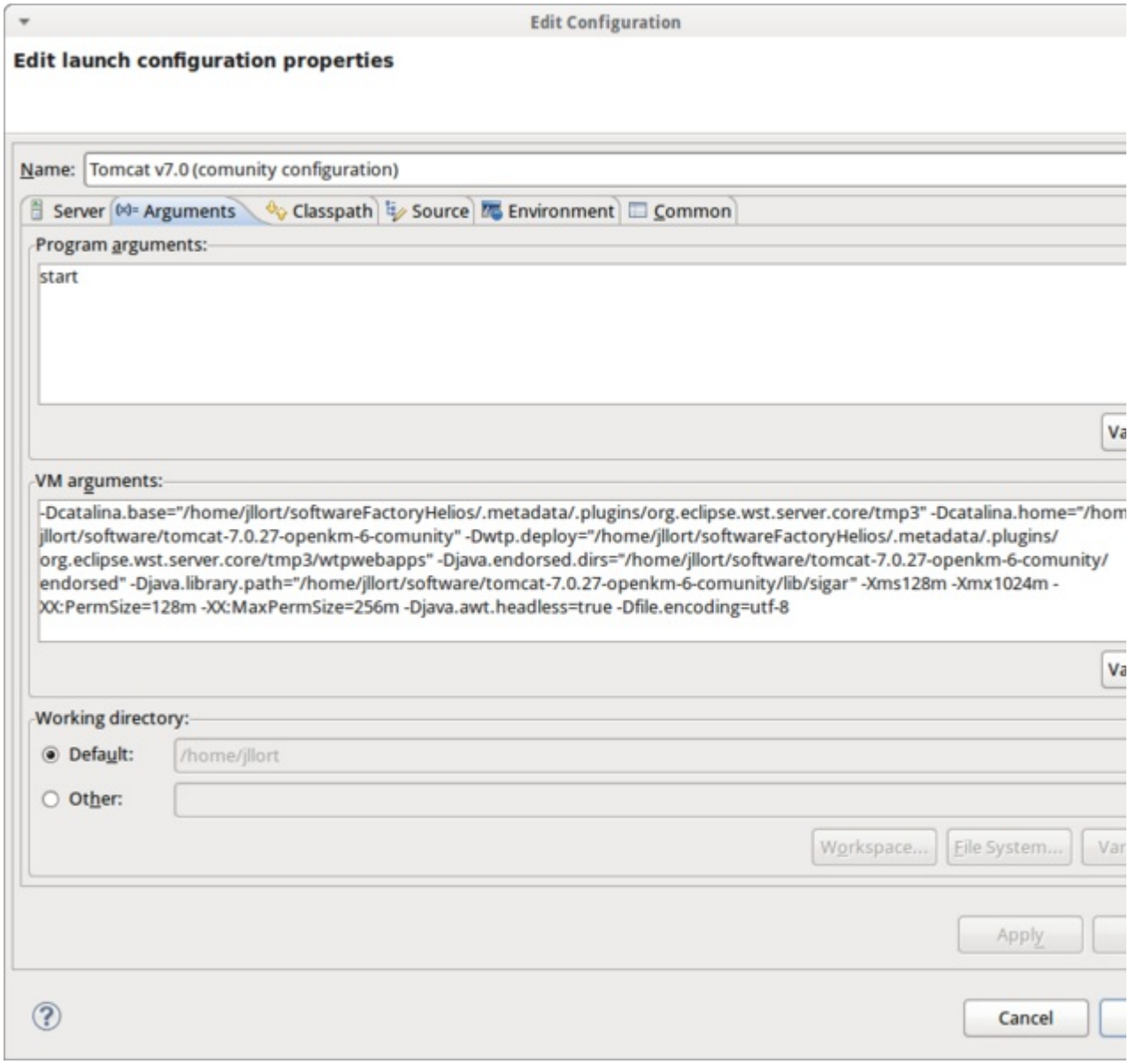
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



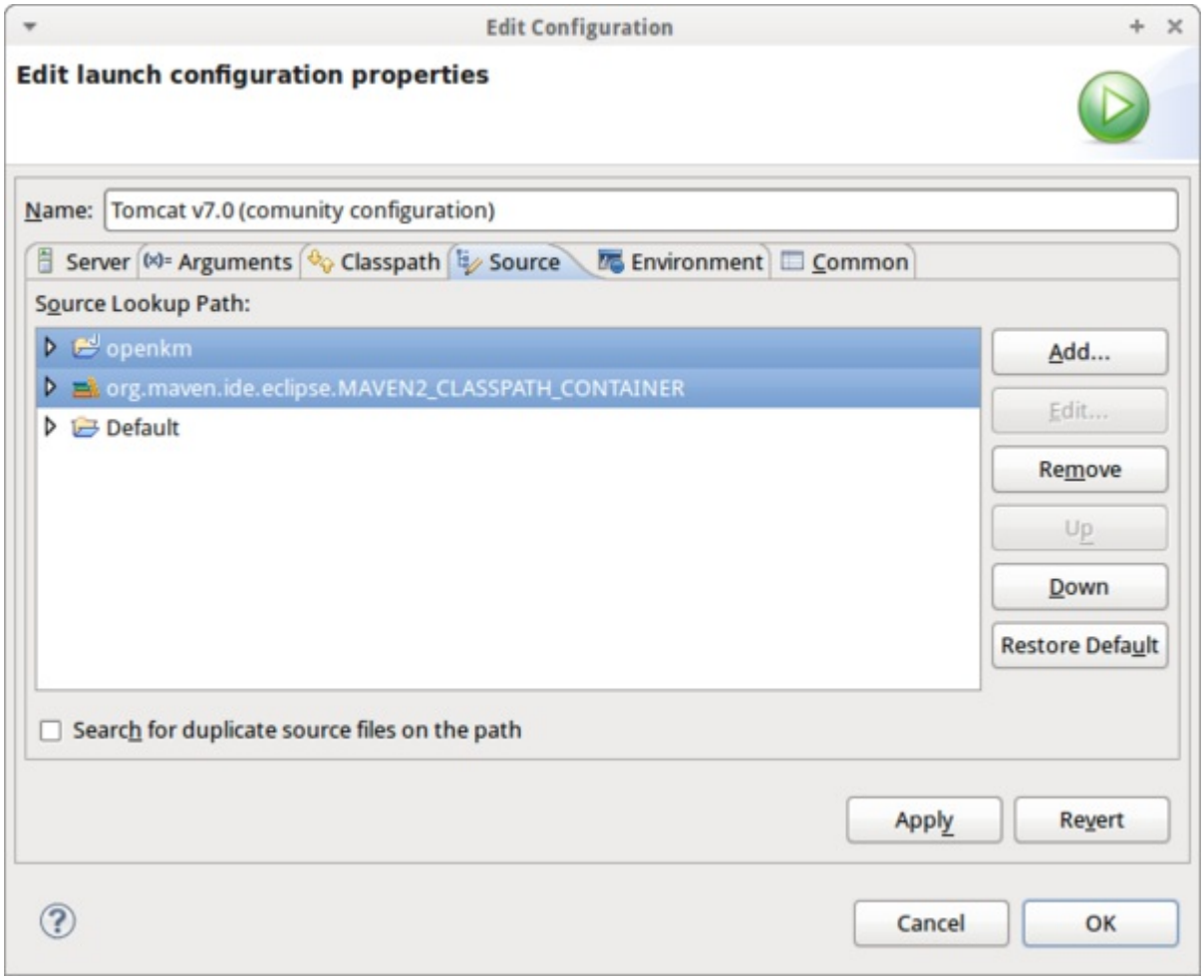
*Edit **launch configuration**: Add **memory parameters** - we suggest 2GB - **sigar** library path and **utf-8** file encoding at VM arguments field:*

```
-Xms128m -Xmx2048m -Djava.awt.headless=true -Dfile.encoding=utf-8 -Djava.library.path
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*Add your **OpenKM project** at source tab:*



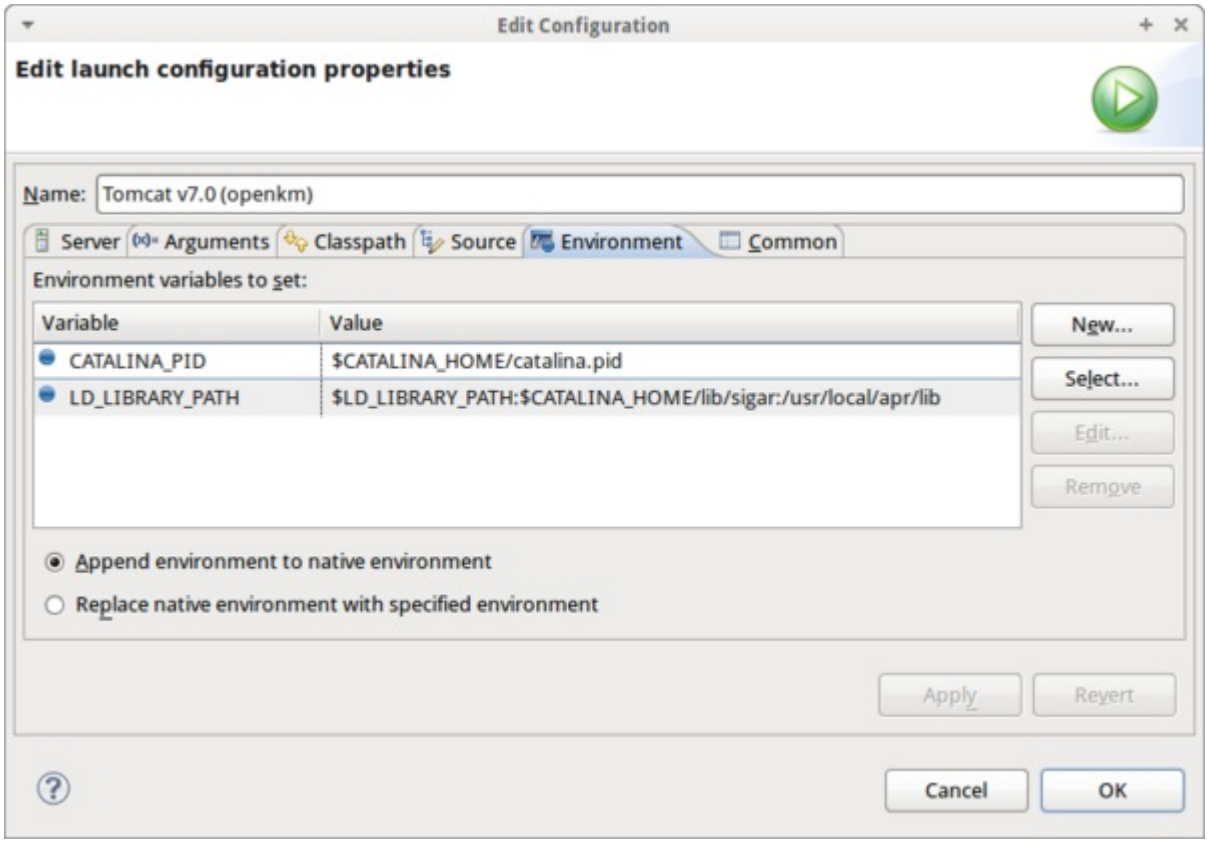
*Add **CATALINA_PID** and **LD_LIBRARY_PATH** at environment variables:*

```
CATALINA_PID=$CATALINA_HOME/catalina.pid
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CATALINA_HOME/lib/sigar:/usr/local/apr/lib
```

or

```
CATALINA_PID=$CATALINA_HOME/catalina.pid
LD_LIBRARY_PATH=/home/jllort/software/tomcat-7.0.27-openkm-6-community/lib/sigar
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Maven support for Eclipse

Latest Eclipse version comes by default with maven support plugins yet installed. For older Eclipse version follow this configuration steps:

- *Go to **Help > Eclipse Marketplace***
- *Install the plugin named M2Eclipse.*

> *Older Eclipse version comes without the "Eclipse Marketplace" feature and the plugins must be installed manually with **Help > New software** option. We encourage do use a more recent version of Eclipse software.*

Subversion support for Eclipse

- *Go to **Help > Eclipse Marketplace***
- *Install the plugin named Subversive.*

> *To checkout a SVN project go to **File > New > Others > Maven > Check out Maven Projects from SCM** .*
>
> *At the beginning the SCM URL list will be empty, you must click on right bottom "**m2e Marketplace**" link and install "**m2e-subversive**" connector.*

> *Older Eclipse version comes without the "Eclipse Marketplace" feature and the plugins must be installed*

*manually with **Help > New software** option. We encourage do use a more recent version of Eclipse software.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configuring Maven in your OS

The latest Eclipse version comes with embedded Maven software already installed. We suggest using the default configuration.

Maven is a software tool for Java project management and build automation. It is similar in functionality to the Apache Ant tool, but is based on different concepts. Maven is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

Maven uses a process known as a Project Object Model (POM) to describe the software project being built, its dependencies on other external modules and components, and the build order. It comes with pre-defined targets for performing certain well defined tasks such as compilation of code and its packaging.

More information at http://maven.apache.org.

Install Maven version 3.3.x

Follow the Maven installation guide.

Maven tips

Download sources and javadoc from libraries dependency to improve the developer experience:

```
$ mvn install -DdownloadSources=true  -DdownloadJavadocs=true
```

Ignore test failures in the build process this way:

```
$ mvn install -Dmaven.test.failure.ignore=true
```

To skip tests, you can do so by executing the following command:

```
$ mvn install -Dmaven.test.skip=true
```

Memory problems

To set more memory for Maven in GNU/Linux you can add the Maven executable to the binary path editing the $HOME/.bashrc file and adding this line at the end:

```
export PATH=$PATH:/path/to/maven/installation/bin
export MAVEN_OPTS="-Xmx512m"
```

On windows set from general environment variables feature.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Configure eclipse to use local Maven

*Go to **Window > Preferences > Maven > Installation** and click on **Add button** to set your local Maven.*

We do not suggest you to apply this change, default embedded Maven configuration usually goes right on almost cases.

Eclipse tips

Debug tomcat

You can debug your OpenKM installation using the Tomcat logging facility. This is an useful thing when you have problems with your configuration. Default OpenKM installation tries to log important events like errors and warnings. Is possible to change this configuration editing the file $TOMCAT_HOME/conf/logging.properties.

Default Tomcat log configuration can generate a lot of messages. These files are stored at $TOMCAT_HOME/logs.

Debug remote tomcat server

When you have problems on the production environment this configuration is specially useful.

Add the following options when the JVM is started:

```
-Xdebug -Xrunjdwp:transport=dt_socket,address=8000,server=y,suspend=n
```

*Or if you are using shell scripts to start Tomcat, start it with the following command (**recommended**):*

```
$ $TOMCAT_HOME/bin/catalina.sh jpda start
```

It will start Tomcat so that a remote debugger can be connected to port 8000.

More information at:

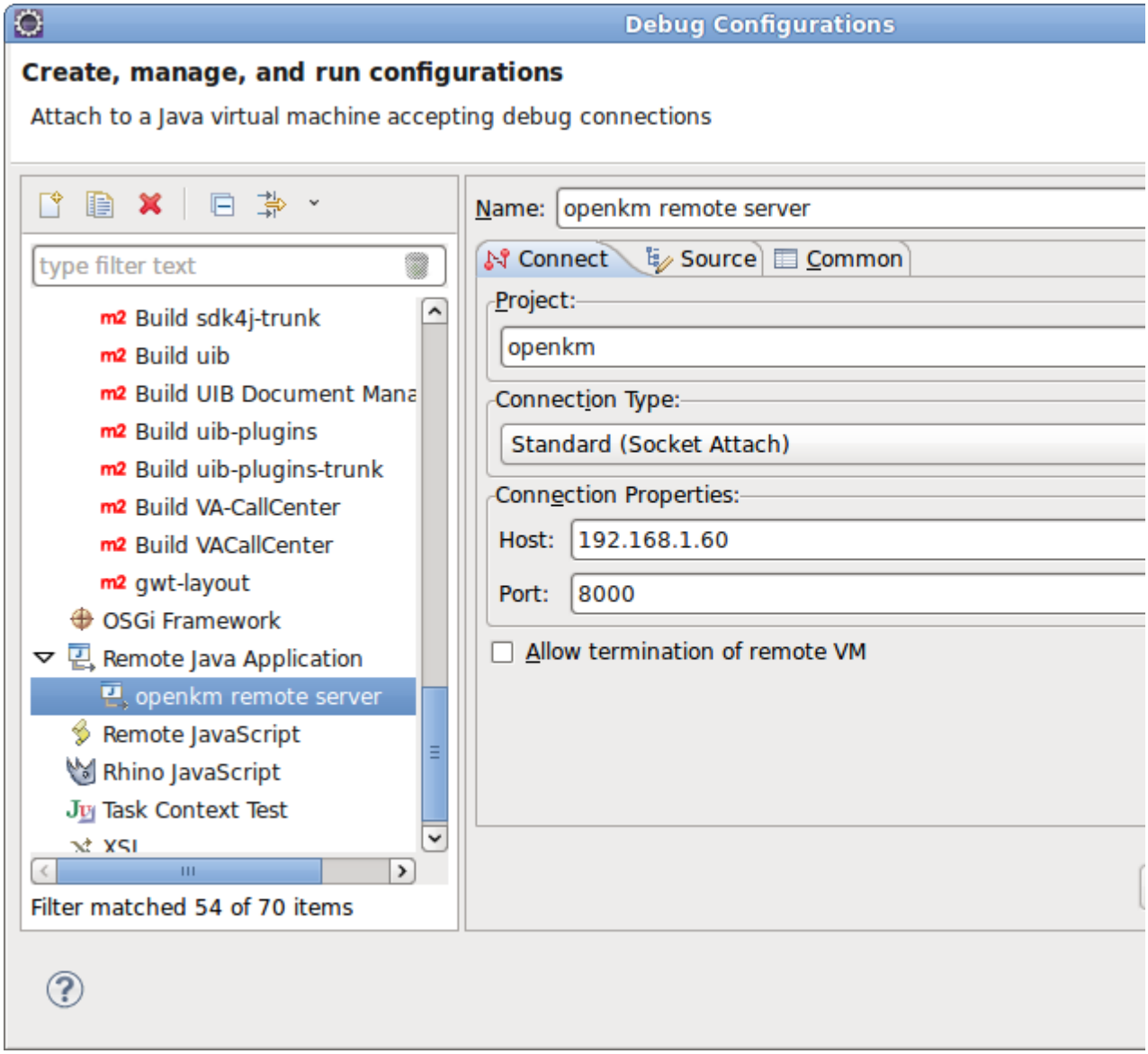
- *Tomcat developing: How do I configure Tomcat to support remote debugging?*
- *Como hacer debug remoto en Tomcat desde Eclipse .*

Eclipse configuration

*Open eclipse "**Debug configurations**", then choose "**Remote Java Application**", right click on mouse and choose "**new**".*

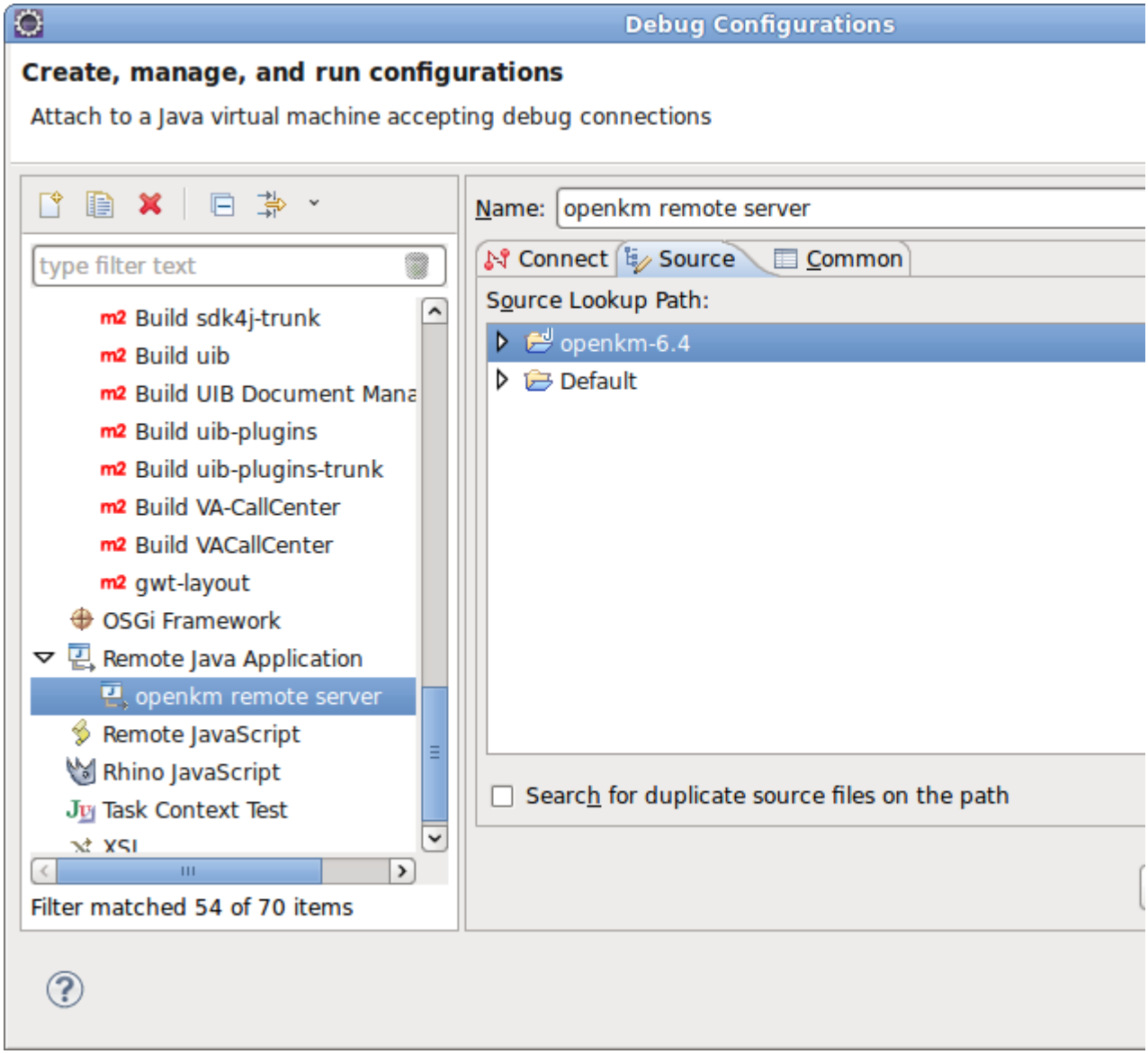
*Set the "**host**" and "**port**" of the remote server:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Include in source tab your OpenKM java project:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



You can debug normally, setting breakpoints, etc.

Debug GWT frontend user interface with GWT Super Dev Mode

> *Super Dev Mode is available from version GWT 2.5.0.RC1, more information at GWT 2.5.0.RC1 Release notes.*
>
> *You can find additional information of them at :*
>
> - *GWT Super Dev Mode offical site.*

Enable source for debugging

*Edit **pom.xml** file and uncomment this section:*

```
<!-- Only for development -->
<dependency>
   <groupId>com.google.gwt</groupId>
   <artifactId>gwt-dev</artifactId>
   <version>${gwt.version}</version>
   <scope>provided</scope>
</dependency>
```

Uncomment the xsiframe section into the Main.gwt.xml file:

```
<!--
  Use GWT 2.7.0 to debug in Firefox -->
  <add-linker name="xsiframe"/>
  <set-configuration-property name='xsiframe.failIfScriptTag' value='FALSE'/>
  <set-configuration-property name="devModeUrlWhitelistRegexp" value="http://(localho
```

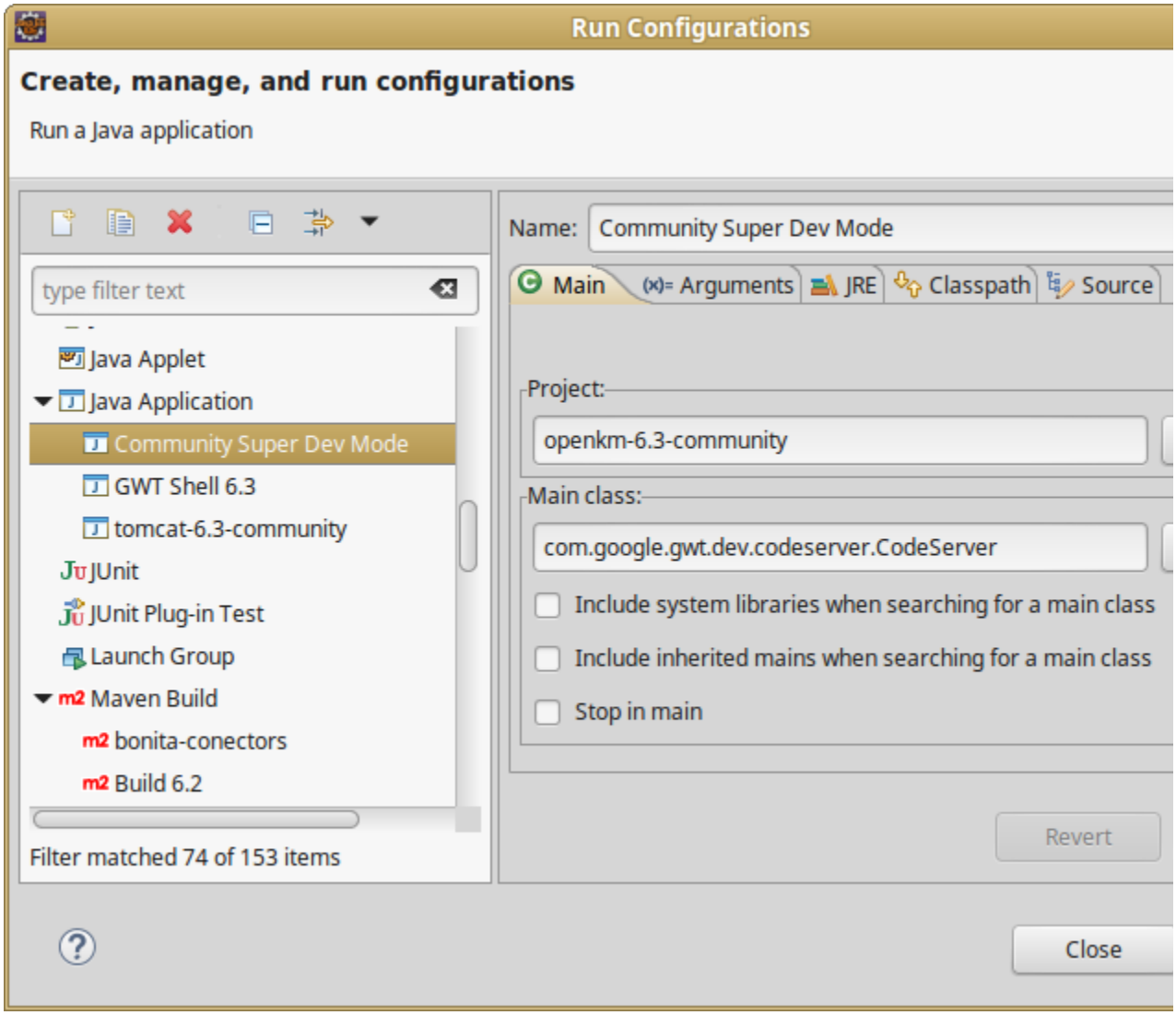
Eclipse configuration

*Open eclipse "**Debug configurations**", then choose "**Java Application**", right click on mouse and choose "**new**".*

Set the Main class field value:

```
com.google.gwt.dev.DevMode
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



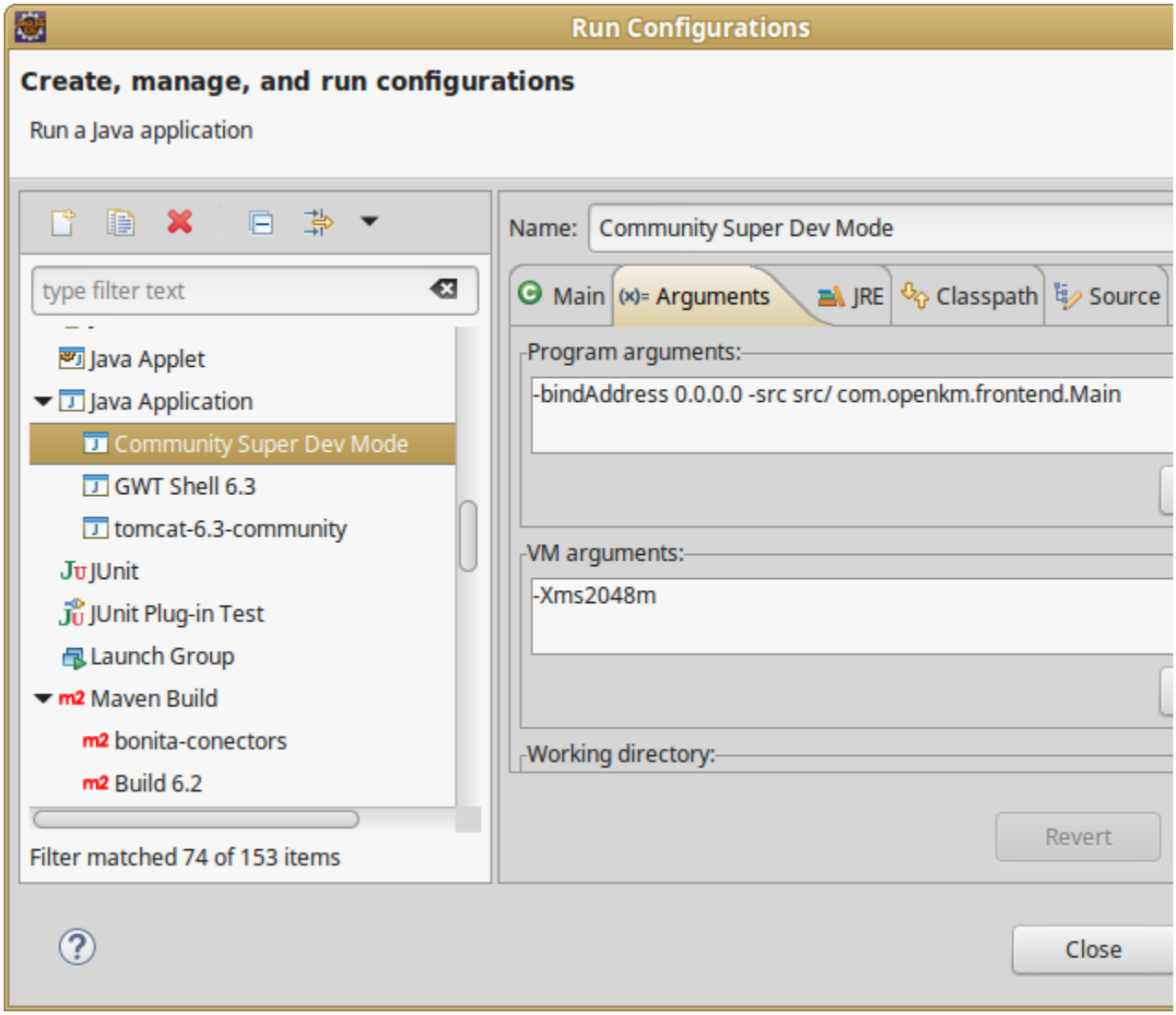
*Select "**Arguments**" tab and set the "**program arguments**" field value:*

```
-bindAddress 0.0.0.0 -src src/ com.openkm.frontend.Main
```

*And the "**VM arguments**" field value:*

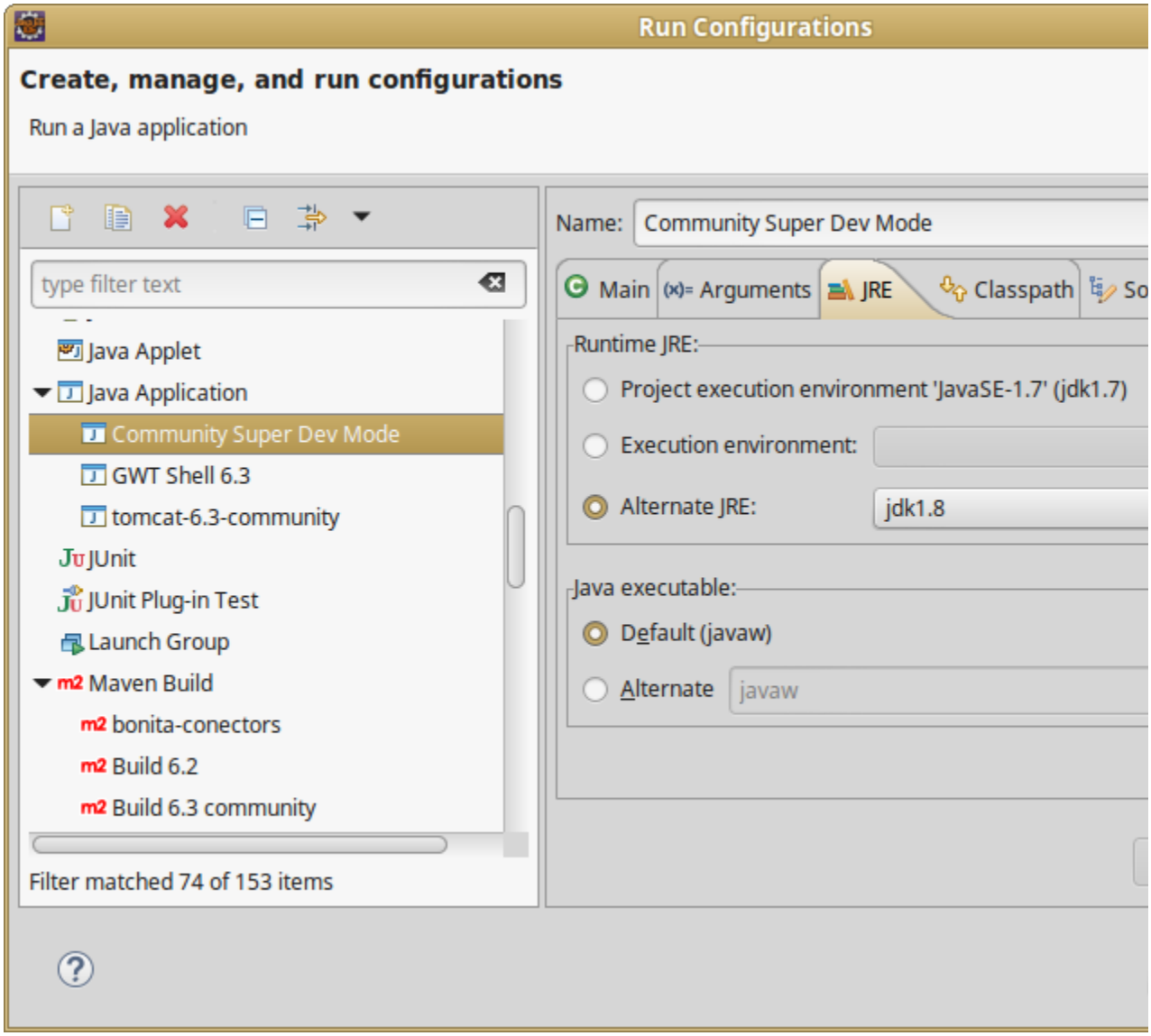
```
-Xms2048m
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



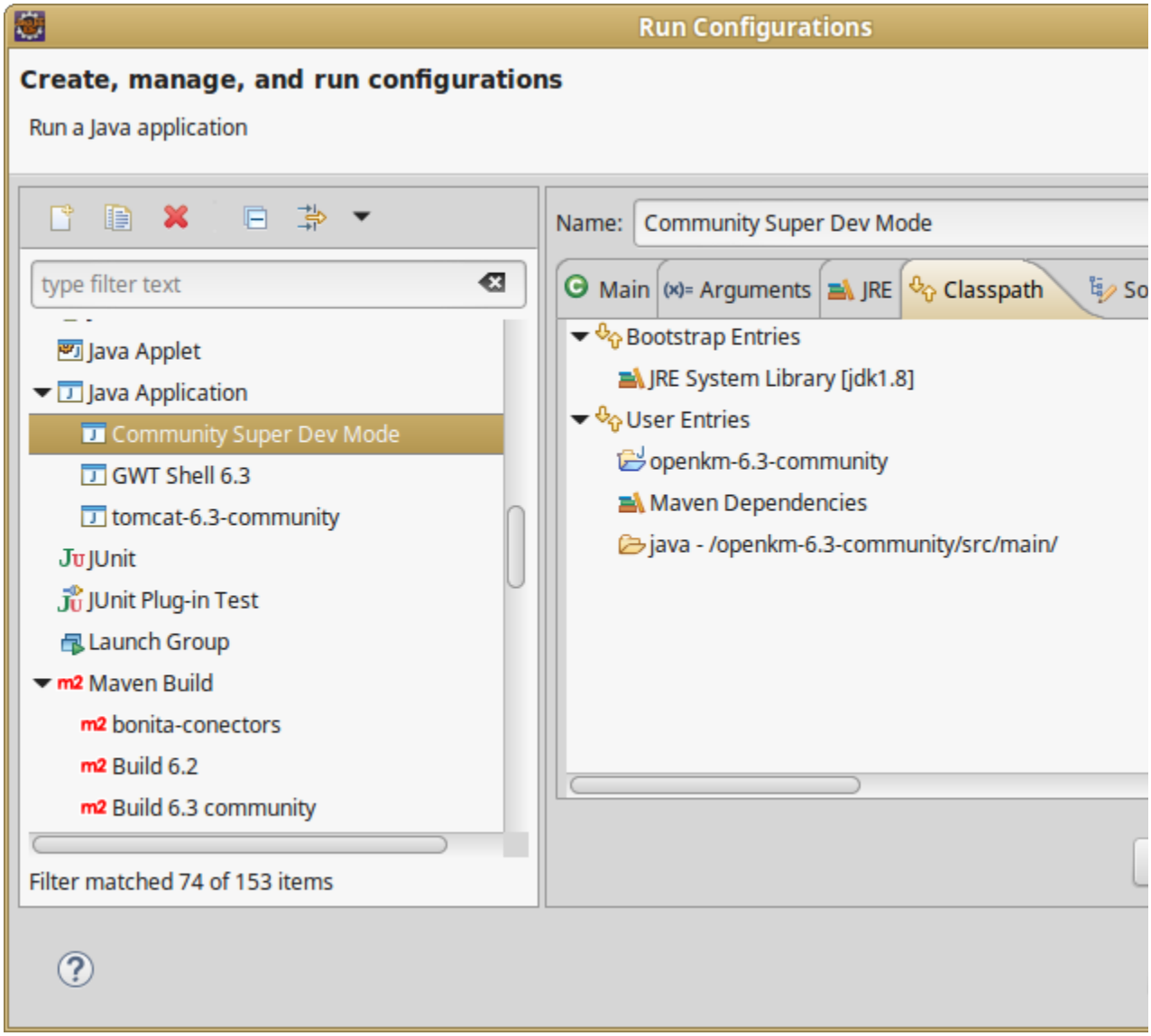
*Select "**JRE**" tab and set the "**OpenJDK version**":*

> *You should choose OpenJDK 1.8.x*



*Select the "**Classpath**" tab and add folder **/openkm/src/main** (use "Advanced" button and "add folder" option for it) :*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

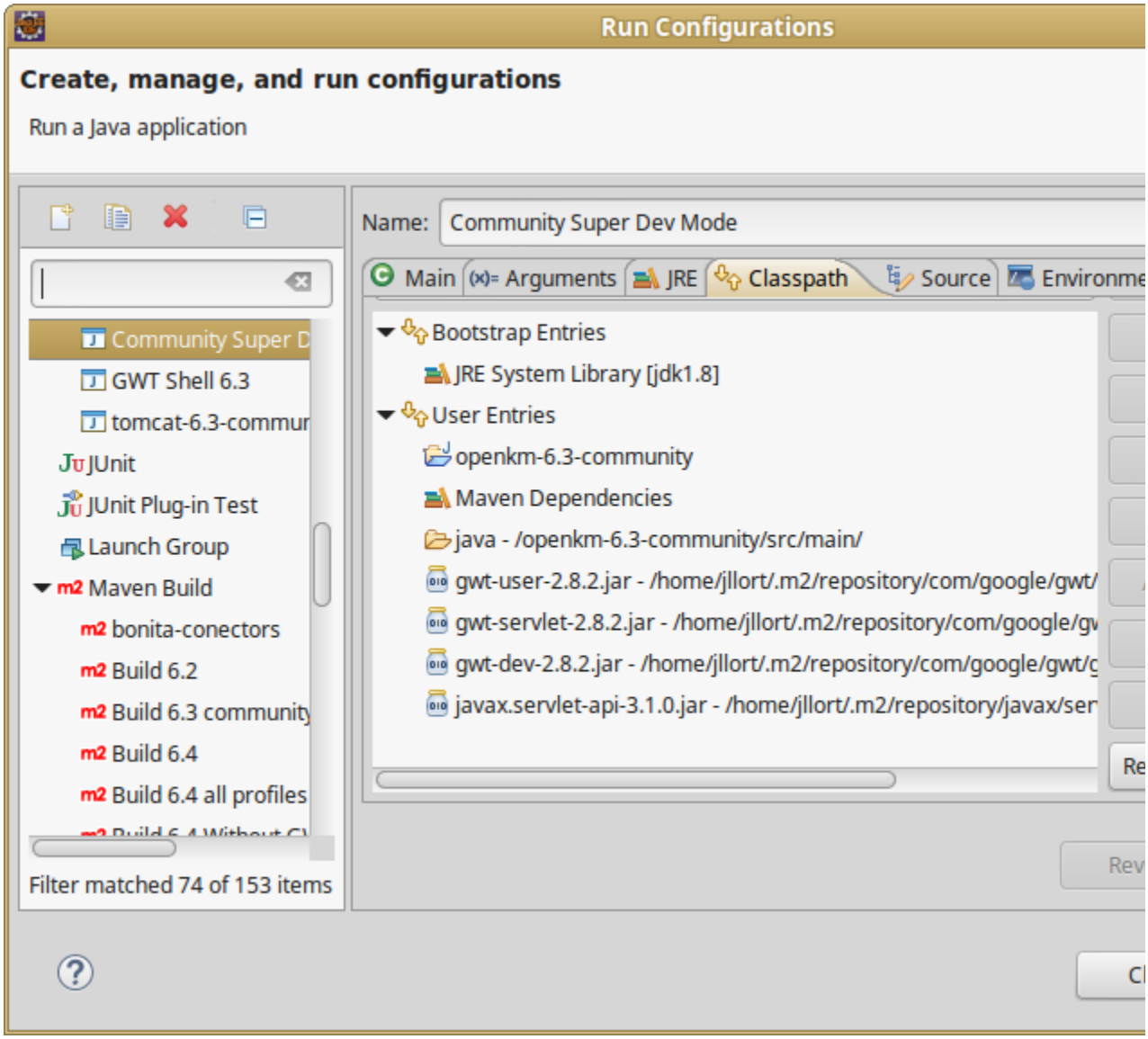


Add GWT jar dependencies with Add External JARs button:

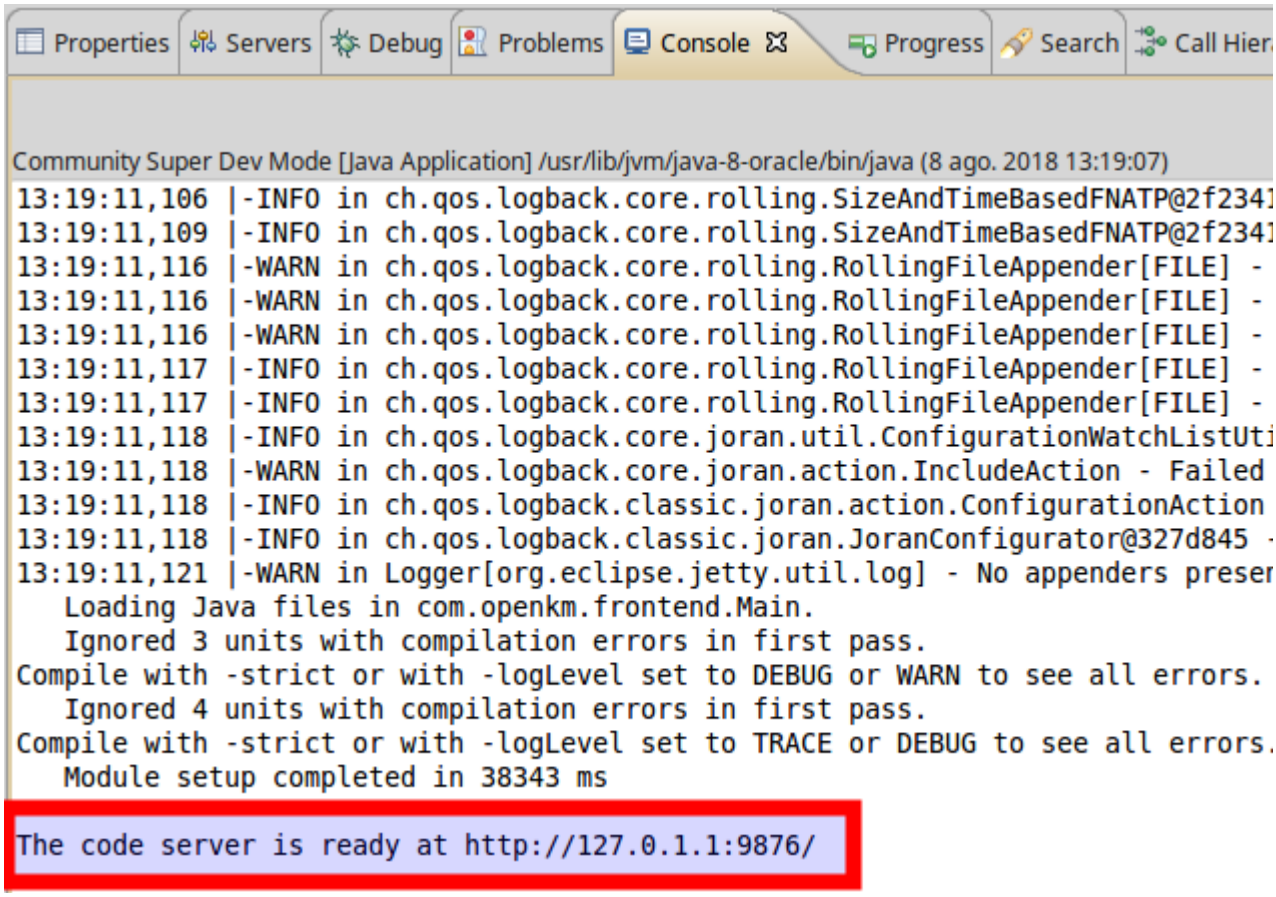
> *The version of JARs files below might change.*
>
> *The javax.servlet-api-3.1.0 solve a java.lang.NoClassDefFoundError: javax/servlet/http/HttpSessionIdListener error. That happens from version 2.8.x and upper.*

- *gwt-user-2.8.2.jar*
- *gwt-dev-2.8.2.jar*
- *gwt-user-2.8.2.jar*
- *javax.servlet-api-3.1.0*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



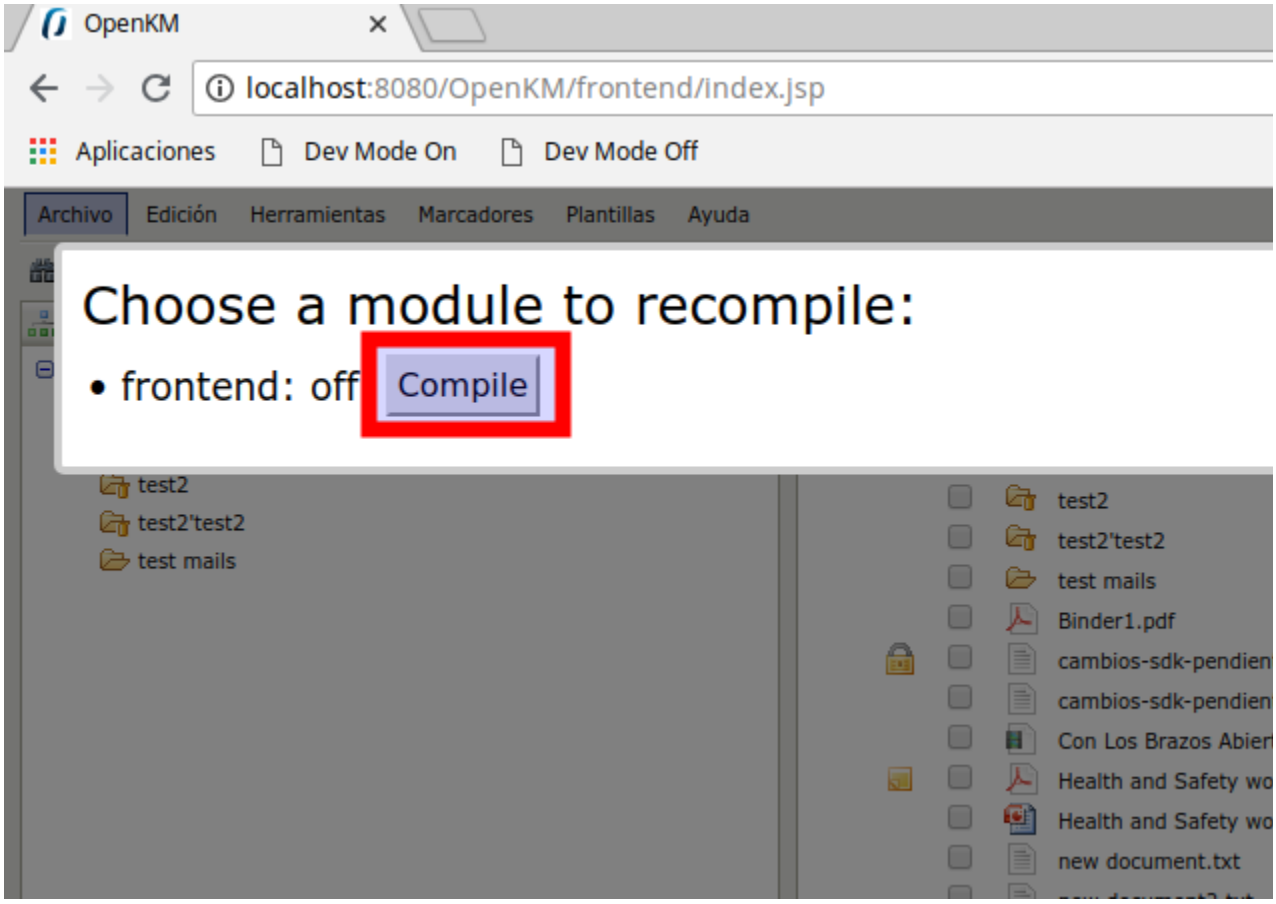
*No you can **run the Super Dev Mode** at the end of the log is shown the url where the service is binding:*



*Open the **code server** url **in your chrome** browser browser **and drag the bookmarklets in your bookmark bar**:*

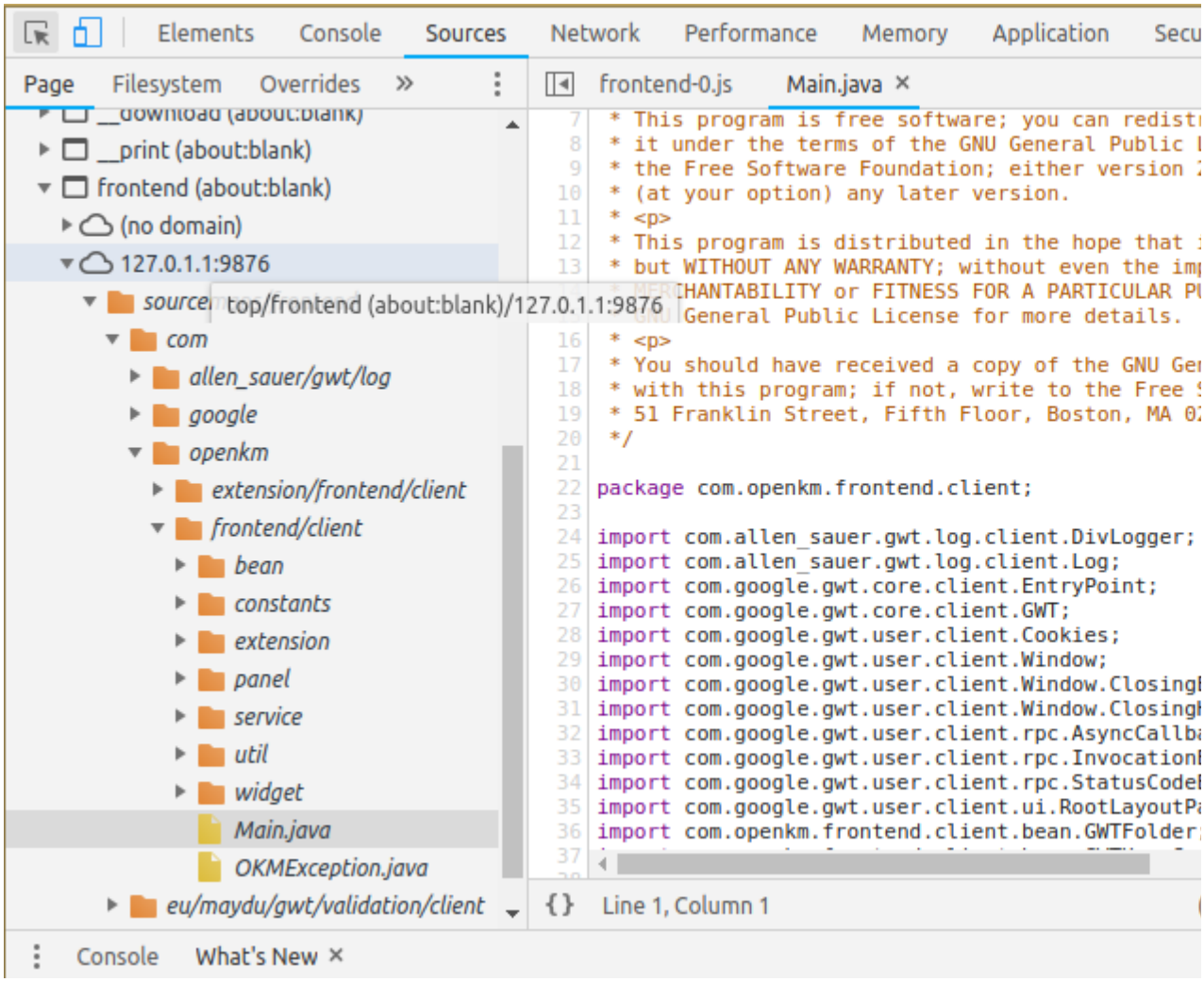
*Open your **OpenKM frontend** url **in** Chorme **browser** and then **click in your bookmark bar the option** "**Dev Mode On**" you have dragged in previous step. It **will be shown a popup option with** Compile **button**. **Click** in "**Compile**" button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*Finally **open** the **Chrome developer tools** and from the **sources tab** you have OpenKM java code available, where to set breakpoints, watch variables, etc:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Debug GWT frontend user interface with GWT Shell

GWT Shell is deprecated an it has been removed from GWT version 2.5.1, more information at GWT 2.5.1 Release notes.

You can debug GWT frontend with GWT Shell (what have extension support from firefox browsers 2.4 and older) or GWT Super Dev Mode (we suggest use chrome in this scenario).

In the documentation section below, we have tried to explain how to get working old GWT Shell with eclipse. The sample below show how to debug GWT from localhost, but also is possible debug remotely.

You can find additional information of them at :

- *GWT Shell official site.*

Enable source for debugging

*Edit **pom.xml** file and uncomment this section:*

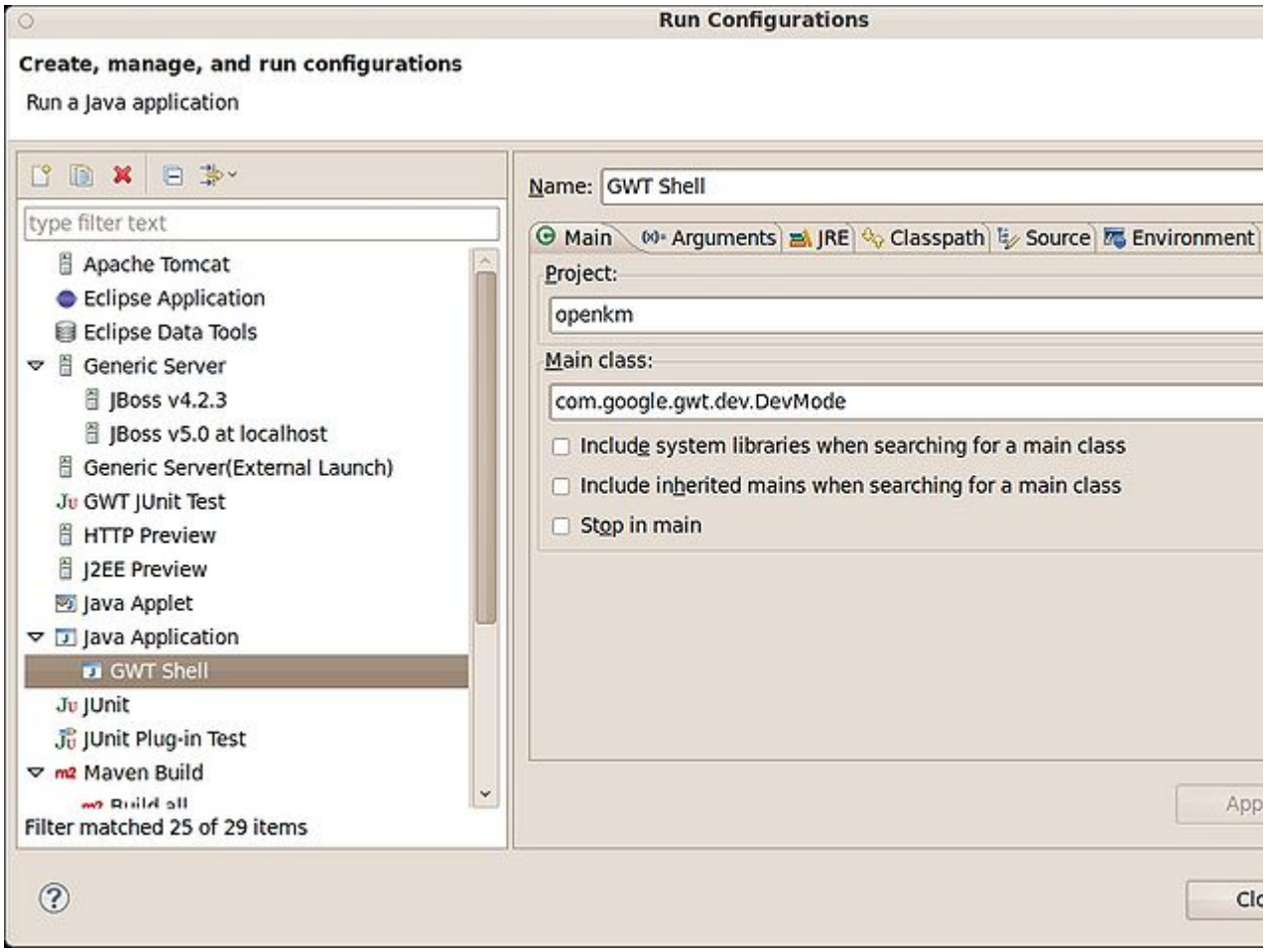
```
<!-- Only for development -->
<dependency>
   <groupId>com.google.gwt</groupId>
   <artifactId>gwt-dev</artifactId>
   <version>${gwt.version}</version>
   <scope>provided</scope>
</dependency>
```

Eclipse configuration

*Open eclipse "**Debug configurations**", then choose "**Java Application**", right click on mouse and choose "**new**".*

Set the Main class field value:

```
com.google.gwt.dev.DevMode
```



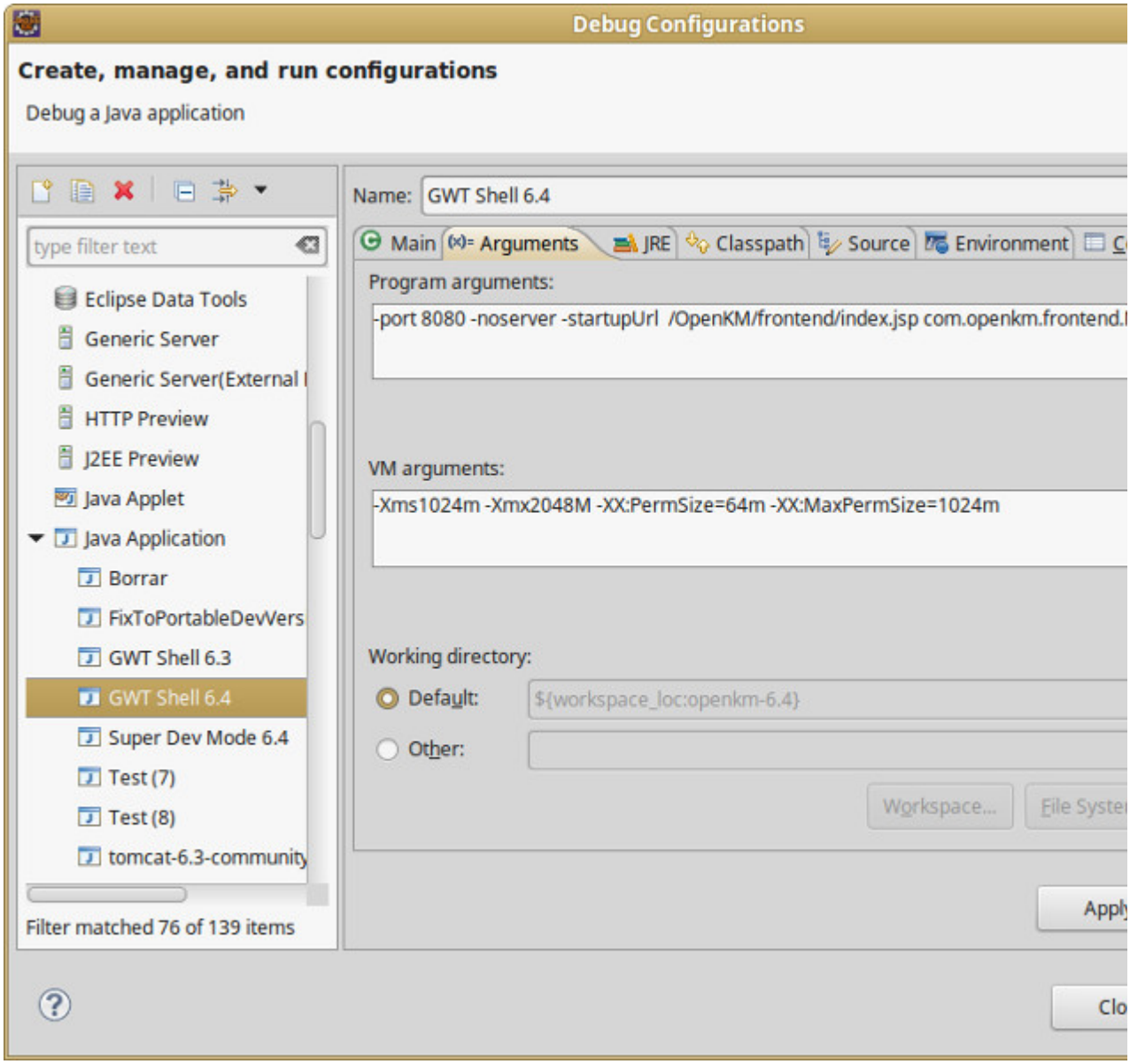
*Select "**Arguments**" tab and set the "**program arguments**" field value:*

```
-port 8080 -noserver -startupUrl  /OpenKM/frontend/index.jsp com.openkm.frontend.Main
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*And the "**VM arguments**" field value:*

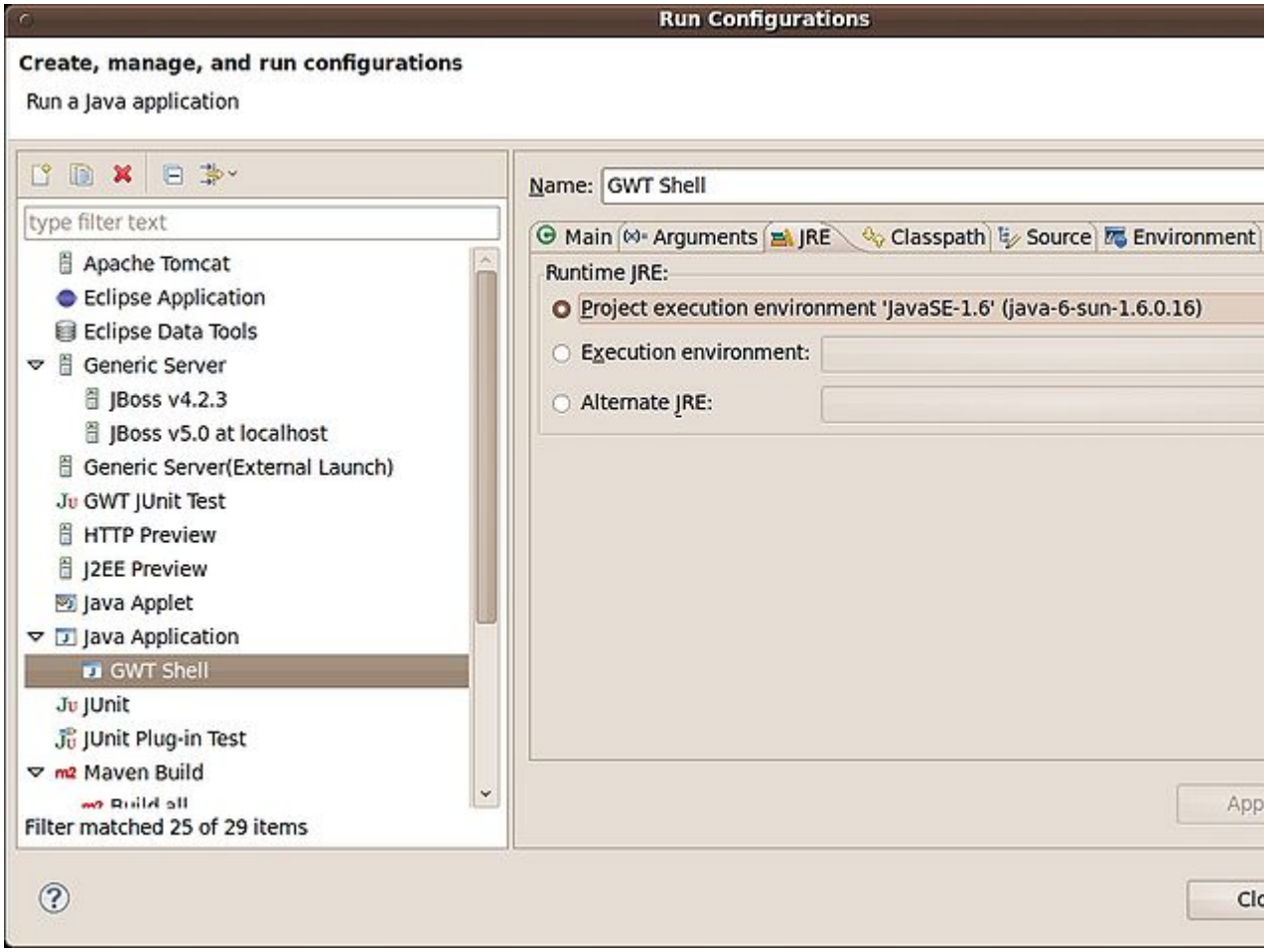
```
-Xms1024m -Xmx2048M -XX:PermSize=64m -XX:MaxPermSize=1024m
```



*Select "**JRE**" tab and set the "**OpenJDK version**":*

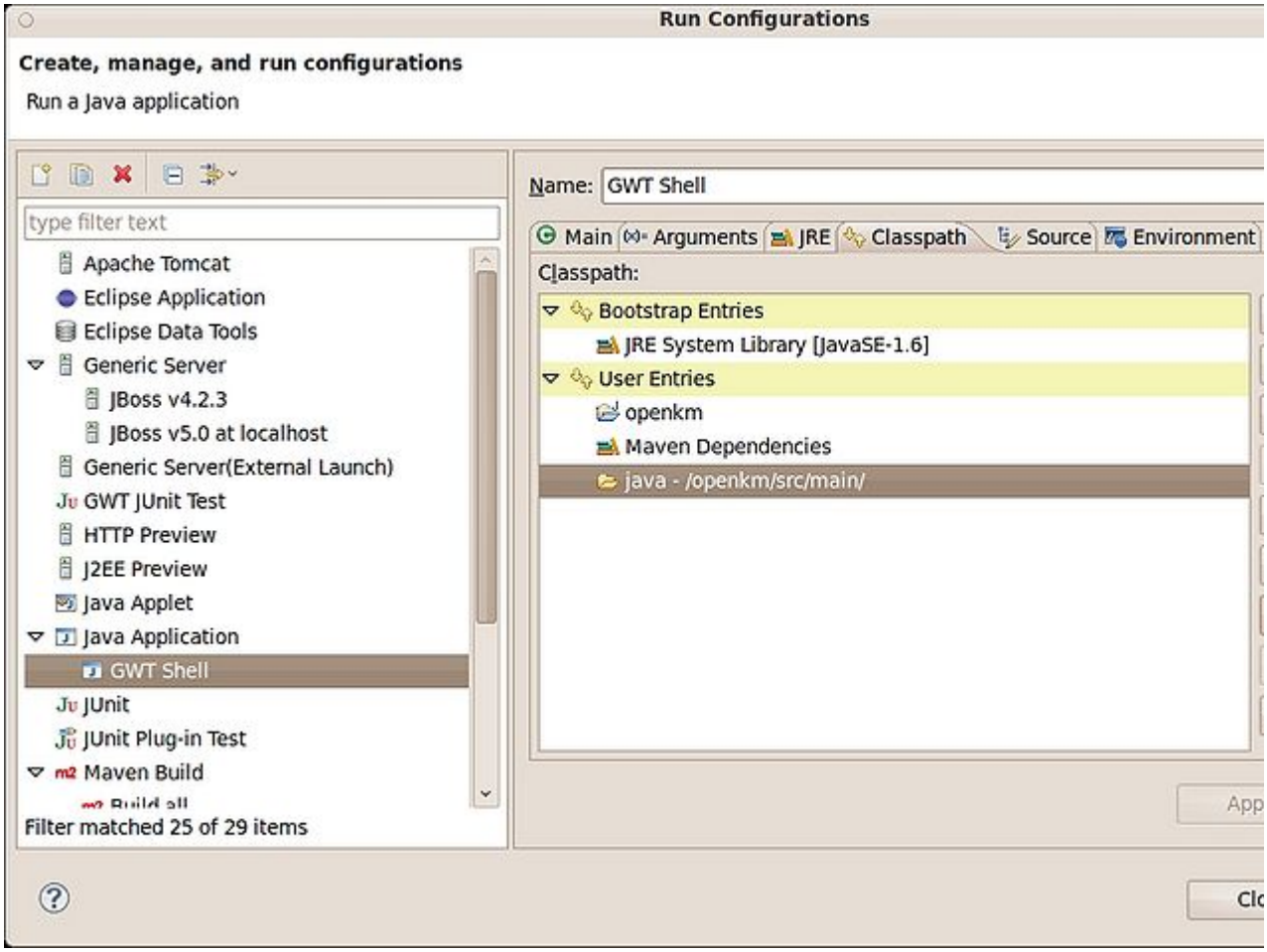
> *You should choose OpenJDK 1.8.x*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*Select the "**Classpath**" tab and add folder **/openkm/src/main** (use "Advanced" button and "add folder" option for it) :*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



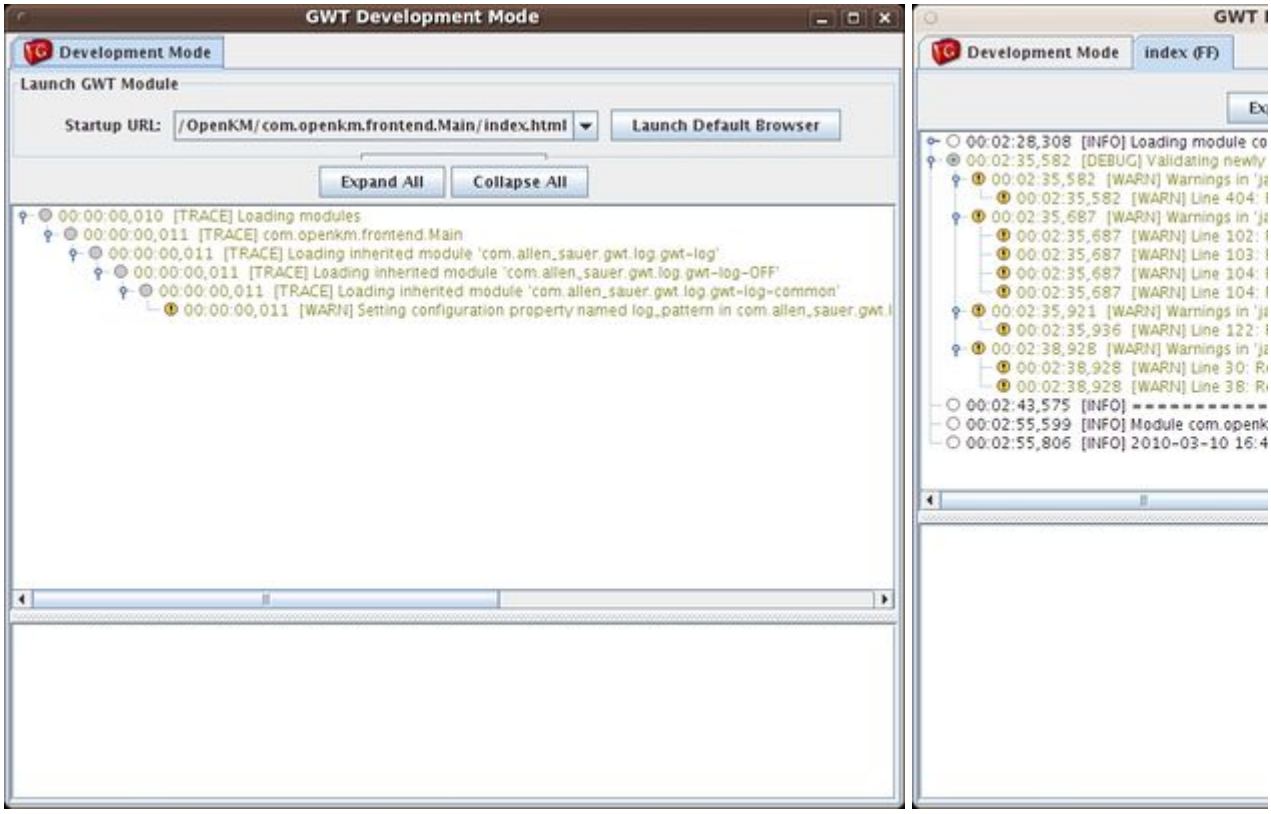
Now you can run the GWT Shell.

*To **debug login** put in your browse:*

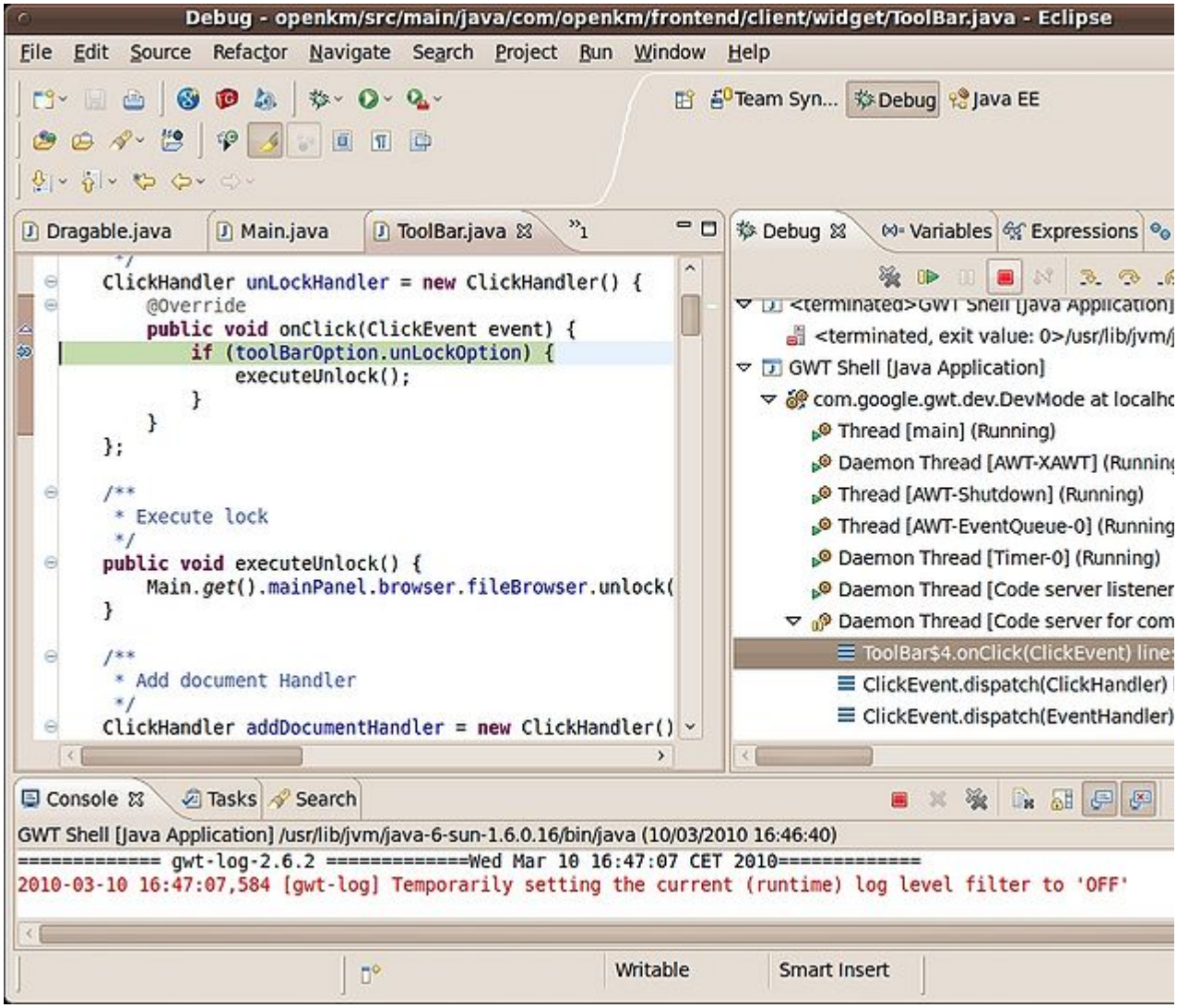
> *You can remotely debug GWT simply changing localhost for your remote URL. Take in consideration you are executing GWT code from eclipse, not the remote one. Consider your local GWT code version be the same is compiled on the remote server.*

```
http://localhost:8080/OpenKM/frontend/index?gwt.codesvr=127.0.1.1:9997
```

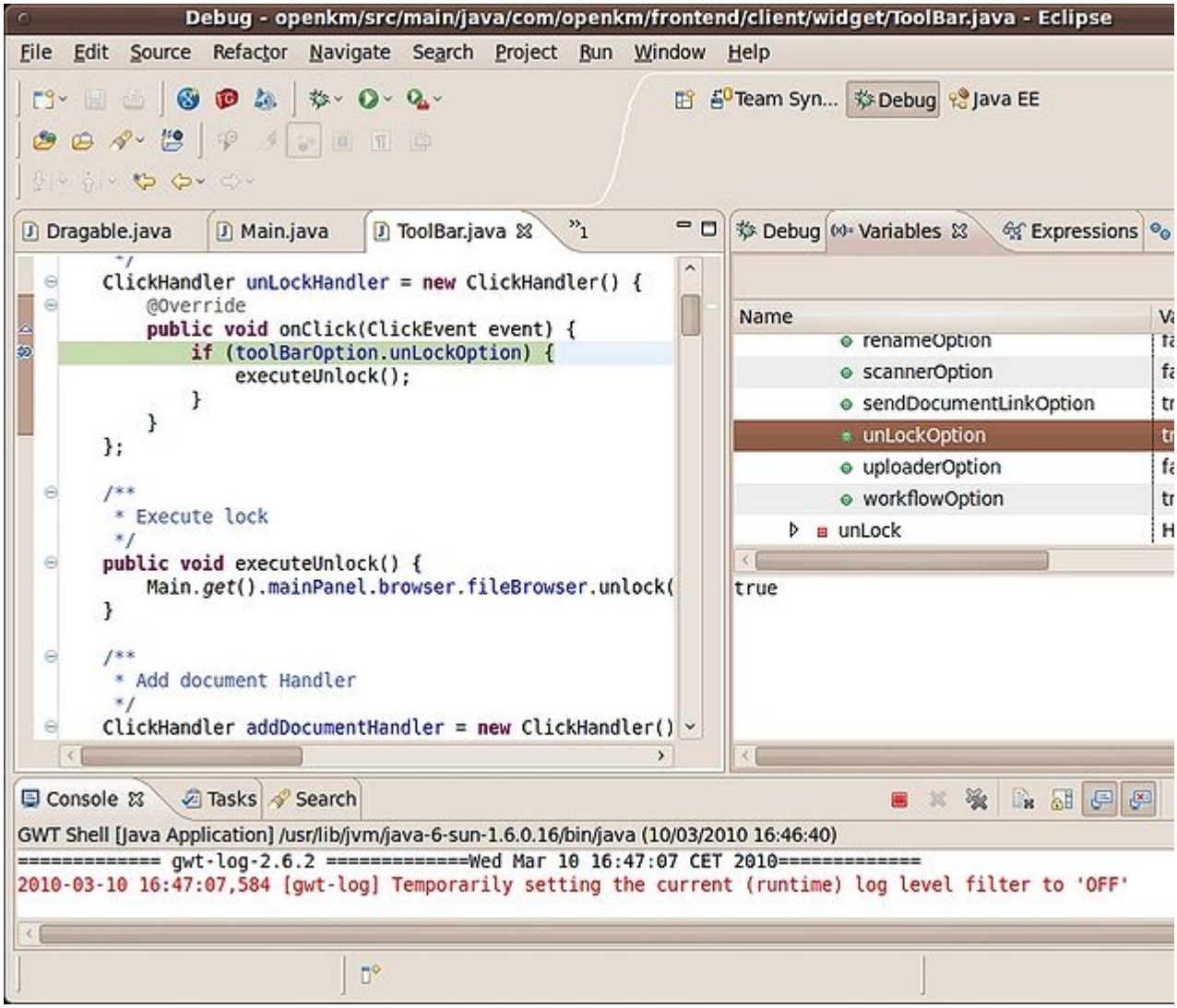
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



You debug into eclipse, see variables values, etc.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Eclipse know troubleshooting

Maven raise error 206 the filename or extension is too long

When you build an application on windows you can get the error ***create process error 206 the filename or extension is too long*** *this is caused to a Windows OS limitation. To solve it:*

- *Move* ***.m2*** *repository to* ***c:***\
- *Go to* ***Eclipse > Windows/Preferences/Maven/User Settings*** *->* ***Create your own setting.xml*** *with its content:*

```
<settings>
  <localRepository>c:/.m2/repository</localRepository>
</settings>
```

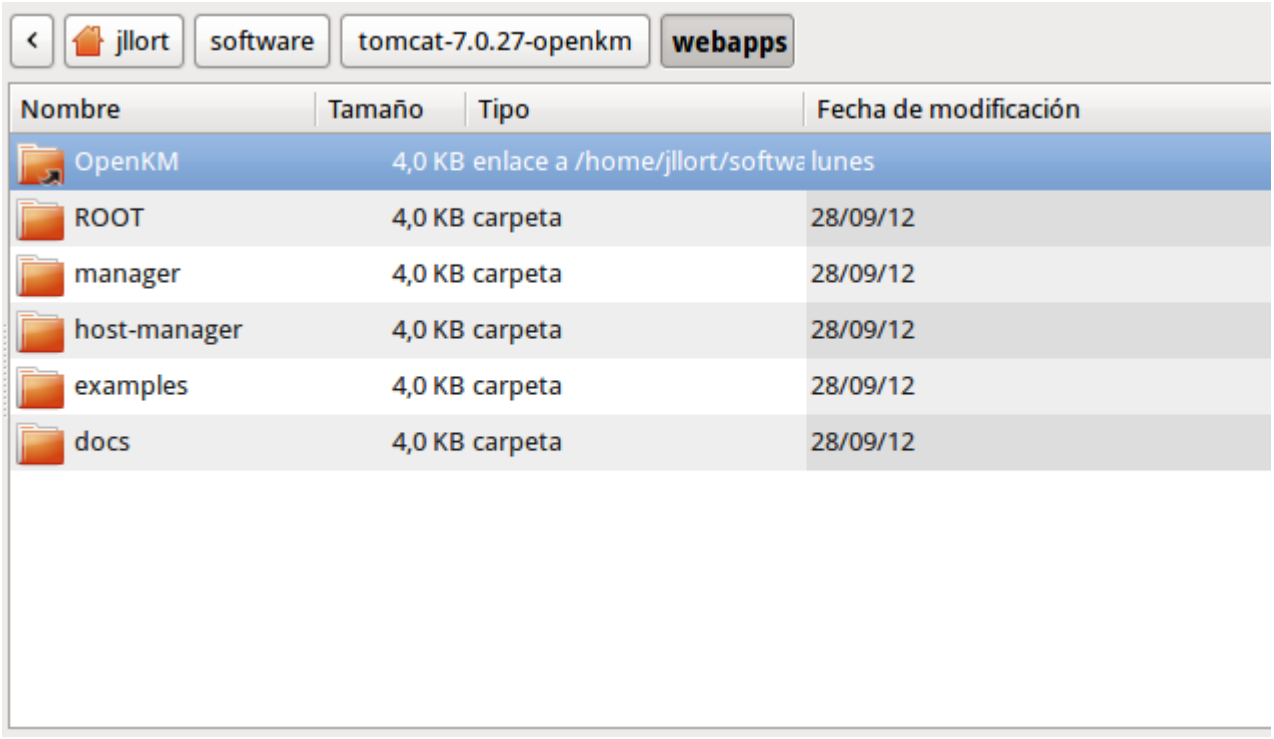
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Development tips

Tomcat application deployment

For faster development we recommend not deploying OpenKM.war directly, file and create in $TOMCAT_HOME/webapps a folder alias of target/OpenKM:

> *To create symbolic links in Windows use "**juntion.exe**" application.*
>
> *More information at http://technet.microsoft.com/en-us/sysinternals/bb896768.*



Compile only GWT frontend module

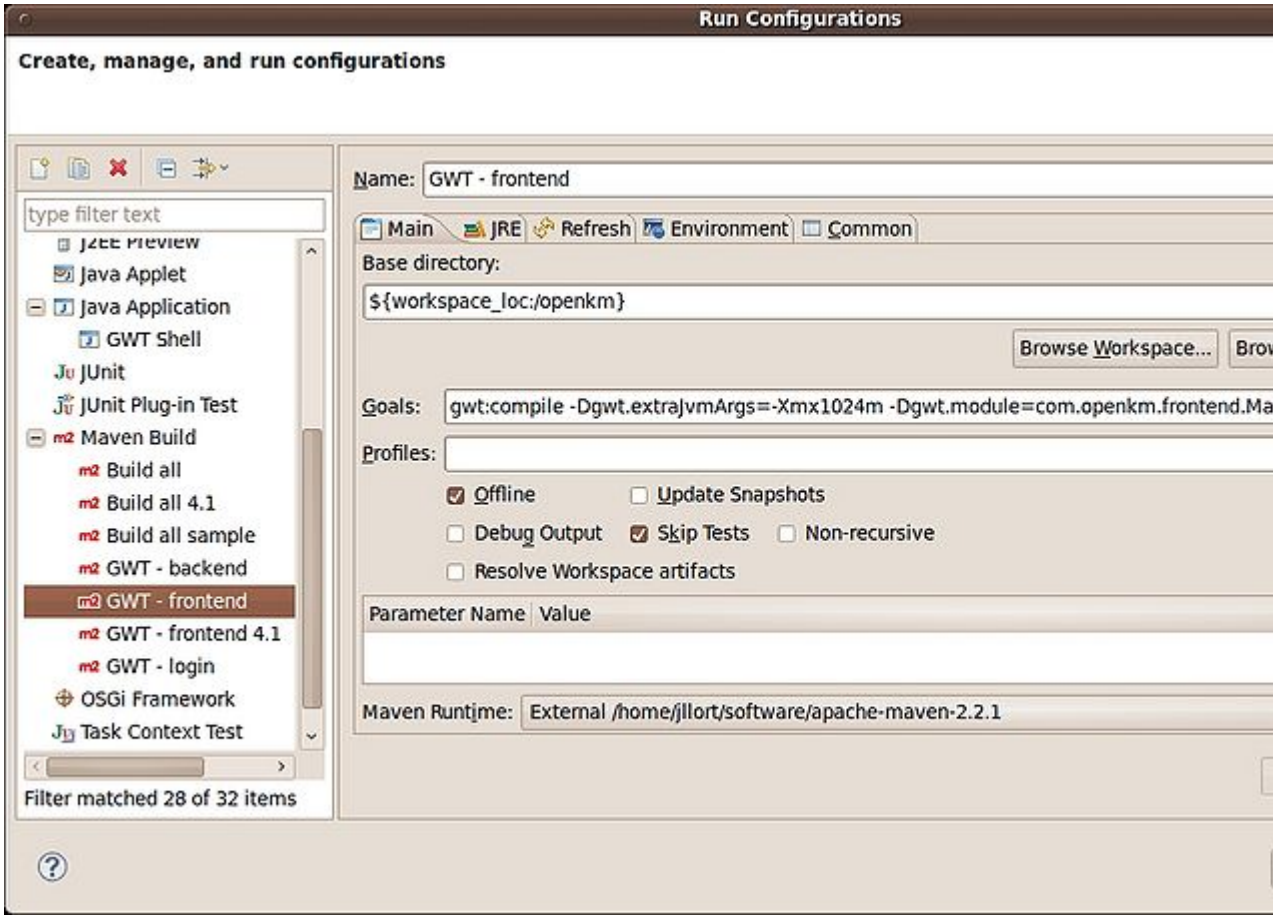
To compile GWT modules individually execute the command:

```
$ mvn gwt:compile -Dgwt.module=com.openkm.frontend.Main
```

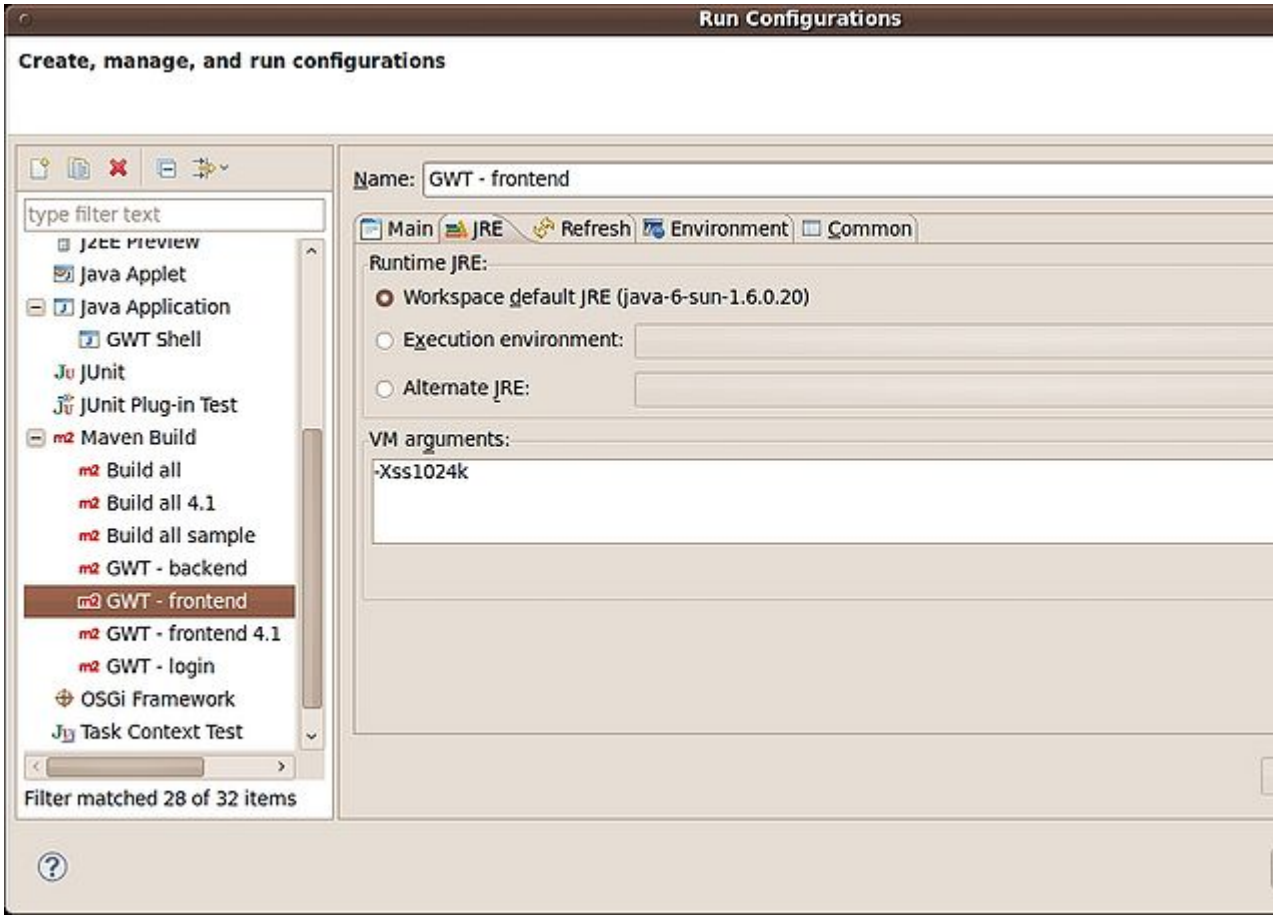
Configure GWT build from eclipse

*Go to "**Run configurations**", choose "**Maven build**" and click on "**right mouse button**" to "**add**" new maven buil.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Compile only one GWT frontend permutation

GWT by default compiles six permutations, each one for distinct browsers. Each permutation consumes a lot of hardware resources (RAM and CPU's).

For fast development and less compilation time enable only a single permutation.

Decreasing the number of enabled permutation, you will decrease the total compilation time needed.

You can also declare several permutations at same time, for example for enabling firefox and chrome can use:

```
<set-property name="user.agent" value="gecko1_8, safari"/>
```

Enable only firefox permutation

Edit the main.gwt.xml OpenKM project file:

```
<!-- Compile for Firefox only -->
<set-property name="user.agent" value="gecko1_8"/>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Enable only chrome permutation

Edit the main.gwt.xml OpenKM project file:

```
<!-- Compile for Firefox only -->
<set-property name="user.agent" value="safari"/>
```

Enable only IE permutation

Edit the main.gwt.xml OpenKM project file:

```
<!-- Compile for Firefox only -->
<set-property name="user.agent" value="ie8"/>
```

Depending the GWT version also are allowed ie6, ie9 and ie10 values.

More information at UserAgent.gwt.xml.

Configure number of CPU's to be used for compiling GWT frontend

Increasing the number of CPU's used by compiling GWT you will decrease the total compilation time needed.

*GWT compilation process can use more than one cpu. You can set the number of cpu to be used with " **localworkers**" parameter.*

***Important**, depending on the number of cores the memory decrease the number of gwt instances compiled at the same time, decreasing locarlWorkers, otherwise your computer may hang.*

*Edit the **pom.xml** project file.*

```
<plugins>
      <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>gwt-maven-plugin</artifactId>
        <version>${gwt.version}</version>
        <executions>
          <execution>
            <goals>
              <goal>compile</goal>
              <!-- <goal>generateAsync</goal> -->
              <!-- <goal>test</goal> -->
            </goals>
          </execution>
        </executions>
        <configuration>
          <runTarget>com.openkm.frontend.Main/index.html</runTarget>
          <modules>
            <module>com.openkm.frontend.Main</module>
          </modules>
          <localWorkers>4</localWorkers>
        </configuration>
      </plugin>
```

In case you're making a lot of changes in OpenKM UI (GWT) we recommend disabling the " ***compile****" directive*

Deploying script sample

You can use a deploying script:

```
#!/bin/bash
# @author: Kenneth Walter

#Cleans and re-compiles the source files
mvn -Dmaven.test.skip=true clean gwt:compile install $*

#Set the TARGET_DIRECTORY to the path of your tomcat installation
TARGET_DIRECTORY=~/Downloads/tomcat-7.0.27/
FILE_TO_MOVE=target/OpenKM.war
#This will only attempt to replace the existing WAR if the new WAR exists
if [ -f $FILE_TO_MOVE ]
then
   echo 'Deploying WAR to Tomcat Directory'
   cp -v $FILE_TO_MOVE $TARGET_DIRECTORY/webapps/.
   echo 'Done'
fi
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Profiling application

You can profile local and remote Java applications using Java VisualVM.

> *For the local application is easy because it detects automatically and show them in the listing. For the remote application you need to do a little work.*

Edit Tomcat startup scripts

*Edit the **$TOMCAT_HOME/bin/setenv.*** files (sentenv.sh for linux and setenv.bat for Windows).*

> *The **java.rmi.server.hostname** should match the IP of the server, usually the **127.0.0.1** value by the right IP.*

```
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote=true"
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.port=9090"
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.ssl=false"
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.authenticate=false"
CATALINA_OPTS="$CATALINA_OPTS -Djava.rmi.server.hostname=127.0.0.1"
```

> *To connect to this remote Tomcat instance:*
>
> - *Go to **File > Add JMX Connection...** and fill the hostname and port.*
> - *Additionally can change the port is 9090 using the **com.sun.management.jmxremote.port** property.*

Authentication

*To enhance security, enable authentication. Edit the **$TOMCAT_HOME/bin/setenv.*** files (sentenv.sh for linux and setenv.bat for Windows).*

```
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.authenticate=true"
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.password.file=$CATALINA_
CATALINA_OPTS="$CATALINA_OPTS -Dcom.sun.management.jmxremote.access.file=$CATALINA_HO
```

*Edit the access allow file **$CATALINA_HOME/conf/jmxremote.access**:*

```
monitorRole readonly
controlRole readwrite
```

> *To show remote threads you need the **readwrite** permission.*

*Edit the password file **$CATALINA_BASE/conf/jmxremote.password**:*

```
monitorRole tomcat
controlRole tomcat
```

Restrict access to these files:

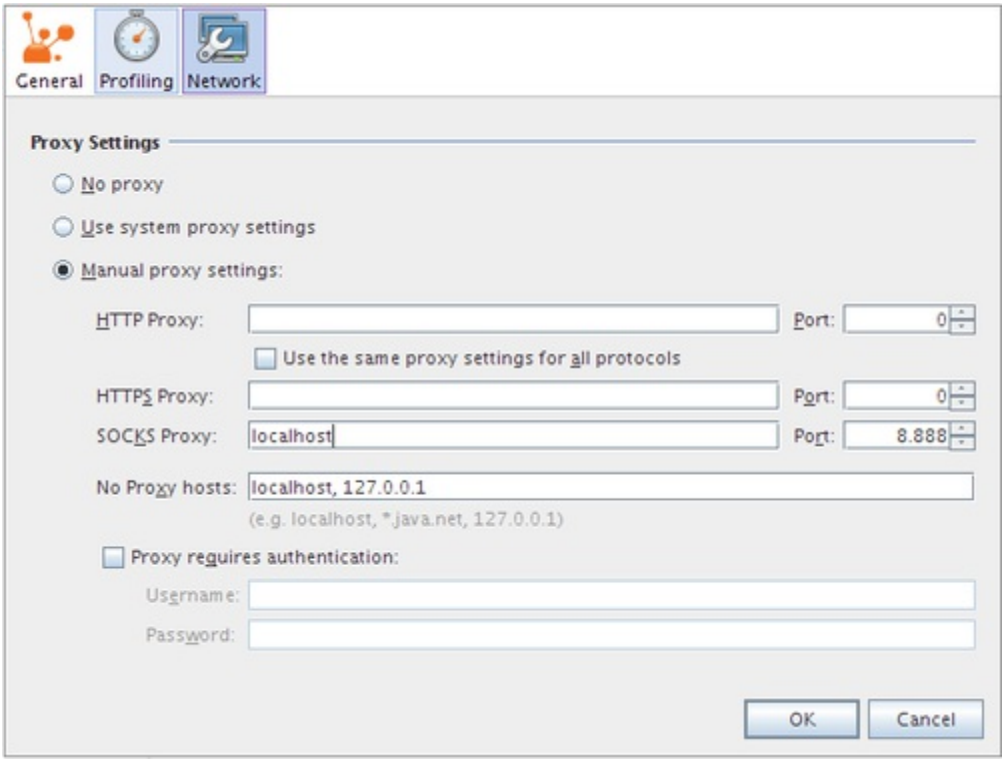
```
$ chmod 600 $CATALINA_HOME/conf/jmxremote.access
$ chmod 600 $CATALINA_HOME/conf/jmxremote.password
```

Monitoring Remote JVM Over SSH

The easiest way to connect to the remote JMX is to use a SOCKS proxy.

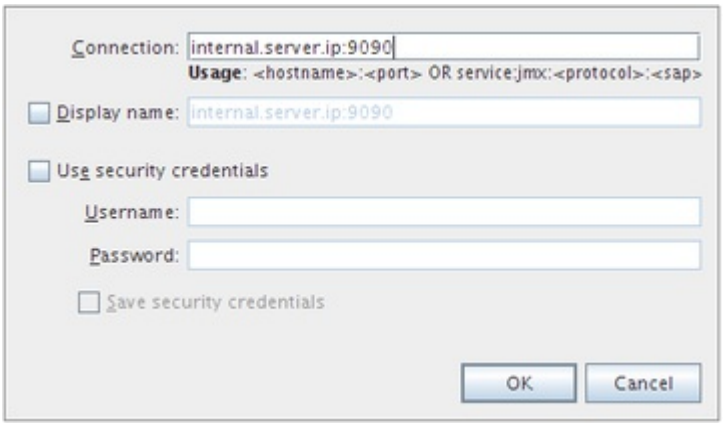
```
$ ssh user@external.server.ip -D 8888
```

Configure VisualVM to use this proxy. Go to ***Tools*** *>* ***Options*** *>* ***Network*** *and edit* ***Manual Proxy Settings****. Configure SOCKS Proxy at localhost and port 8888.*



Now you can connect VisualVM to the remote target. Go to ***File*** *>* ***Add JMX Connection...*** *and type the IP or hostname of the remote machine and the configured JMX port (in our sample configuration is 9090).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



More information at:

- *Profiling With VisualVM, Part 1*
- *Profiling With VisualVM, Part 2*
- *Connecting Visual VM to Tomcat 7*
- *Monitoring and Managing Tomcat 7*
- *Remote Profiling of JBoss using VisualVM*
- *Troubleshooting application performance with VisualVM*
- *Monitoring of Tomcat with VisualVM and VisualGC*
- *Using VisualVM to fix live Tomcat and JVM problems*

HPROF

Java includes HPROF, a profiler which collect application runtime information. HPROF is capable of presenting CPU usage, heap allocation statistics, and monitor contention profiles.

For example, can collect CPU usage information by sampling threads.

*Edit **$TOMCAT_HOME/bin/setenv.sh** file (or **setenv.bat** if using Windows):*

```
CATALINA_OPTS="$CATALINA_OPTS -agentlib:hprof=cpu=samples"
```

*When Tomcat starts you can see a file called **java.hprof.txt**. The CPU profiling info will be dumped to this file once Tomcat process is stopped.*

More information at:

- *HPROF: A Heap/CPU Profiling Tool in J2SE 5.0*
- *HPjmeter*

Application version compatibility table

Application compatibility changes can be caused by:

- *DTD definition changes. Usually, newer application version supports older DTD versions.*
- *Libraries upgrades.*

If your OpenKM server can't access to the Internet, you can download this DTD and copy to a convenient place at your server. Remember to update your forms.xml with the TDT location.

```
<!DOCTYPE workflow-forms PUBLIC "-//OpenKM//DTD Property Groups 3.8//EN"
                                "file:///path/to/property-groups-3.8.dtd">
```

Compatibility tables:

Property Groups	*OpenKM*
property-groups-3.8.dtd	*Since 7.1.9*
property-groups-3.7.dtd	*Since 7.1.5*

Workflow Forms	*OpenKM*
workflow-forms-2.12.dtd	*Since 7.1.18*
workflow-forms-2.11.dtd	*Since 7.1.5*

Report Parameters	*OpenKM*
report-parameters-2.1.dtd	*Since 7.1.5*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Change default login screen

Configuring Tomcat external resource

- Stop the Tomcat.
- Edit the "**$TOMCAT/conf/server.xml**" file and add into xml "**<Host>...</Host>**" section add the line:

```
<Context docBase="${catalina.base}/custom" path="/openkm/custom" reloadable="true"/>
```

The xml Host sections should looks like this:

```
<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true">

 <!-- SingleSignOn valve, share authentication between web applications
   Documentation at: /docs/config/valve.html -->
 <!--
 <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
 -->

 <!-- Access log processes all example.
   Documentation at: /docs/config/valve.html
   Note: The pattern used is equivalent to using pattern="common" -->
 <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
     prefix="localhost_access_log." suffix=".txt"
     pattern="%h %l %u %t &quot;%r&quot; %s %b" />


 <!-- External resources -->
 <Context docBase="${catalina.base}/custom" path="/openkm/custom" reloadable="true"/>
</Host>
```

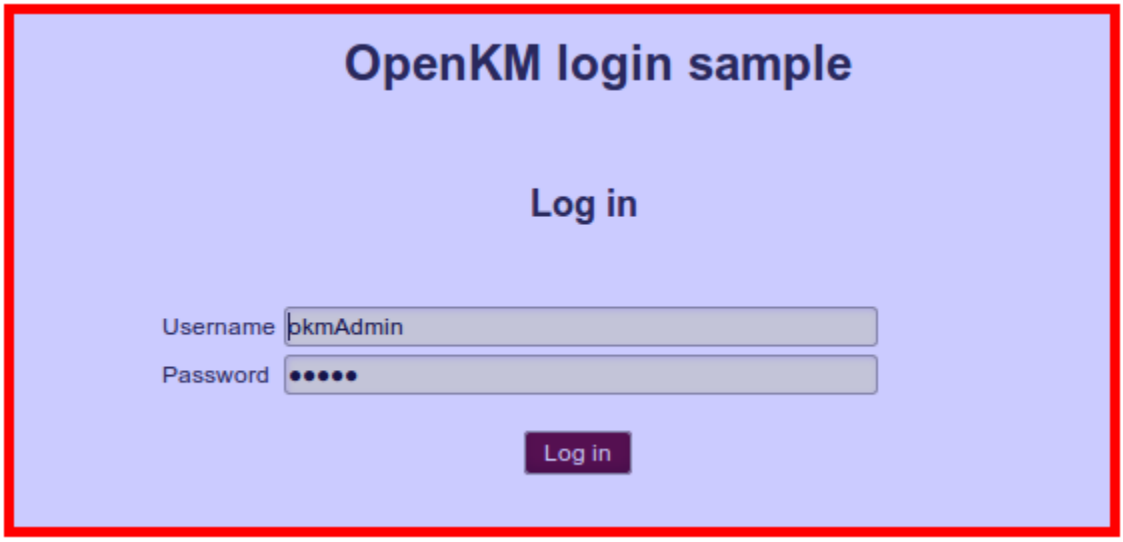
- Unzip the custom.zip file.
- Start Tomcat.

Change default login web page

Go to **Administration** > Configuration parameters and edit the parameter:

Property	Type	Description
custom.login	**String**	`/custom/login.jsp`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

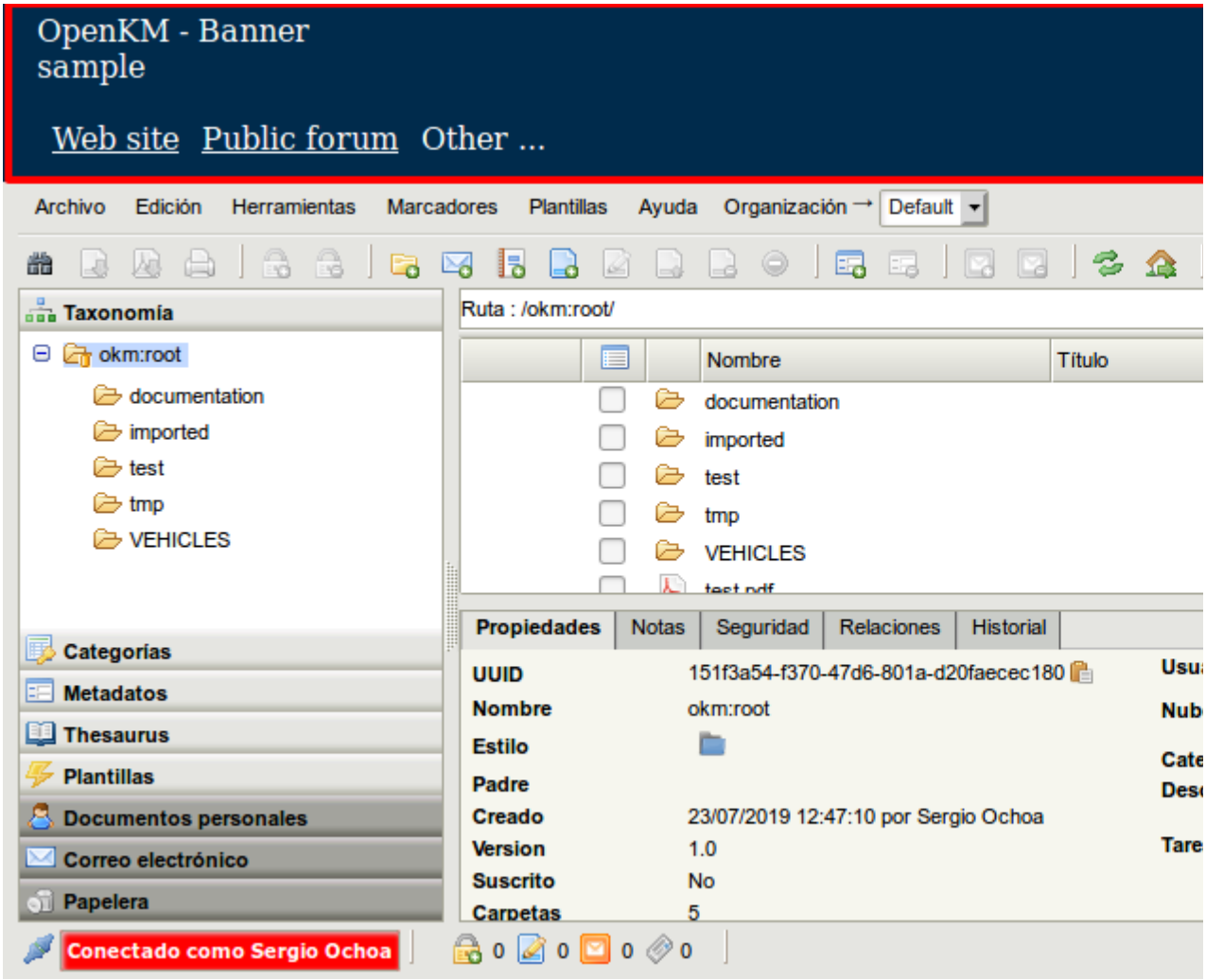


Enable frontend top banner section

*Go to **Administration** > Configuration parameters and edit the parameters:*

Property	*Type*	*Description*
custom.header.height	***Integer***	`95`
custom.header.source	***String***	`/openkm/custom/header.jsp`

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


OpenKM - Banner sample
Web site
Public forum
Other ...
Archivo
Edición
Herramientas
Marcadores
Plantillas
Ayuda
Organización
Default
Taxonomía
okm:root
documentation
imported
test
tmp
VEHICLES
Categorías
Metadatos
Thesaurus
Plantillas
Documentos personales
Correo electrónico
Papelera
Conectado como Sergio Ochoa
Ruta : /okm:root/
Nombre
Título
documentation
imported
test
tmp
VEHICLES
test.pdf
Propiedades
Notas
Seguridad
Relaciones
Historial
UUID
151f3a54-f370-47d6-801a-d20faecec180
Nombre
okm:root
Estilo
Padre
Creado
23/07/2019 12:47:10 por Sergio Ochoa
Version
1.0
Suscrito
No
Carpetas
5

Change folder style by metadata value

The example demostrates how to change folder style icons based on metadata values.

When a document into a folder changes their status value it cause the change of the parent folder icon.

Configure sample

Create a new Metadata group

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
 <property-group label="Invoice status" name="okg:invoicedata">
  <select label="Status" name="okp:invoicedata.status" type="simple">
      <option label="Open" value="open" />
      <option label="Pending to review" value="review" />
      <option label="Closed" value="closed" />
  </select>
 </property-group>
</property-groups>
```

More information about Metadata.

Create new Folder styles

Create three new folder styles. Each folder style will use only a single icon for all the status.

Folder style name	Icon	Internal Id
green		2
orange		3
red		4

More information about Folder style.

Each folder style internaly have and unique Id. That Id is unique an automatically assigned by application on creation stage. On the sample has been used 2,3,4 unique Id but on your application can be others.

To know what are your Id put your cursor over the Edit folder style icon and copy the url. You'll see something

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

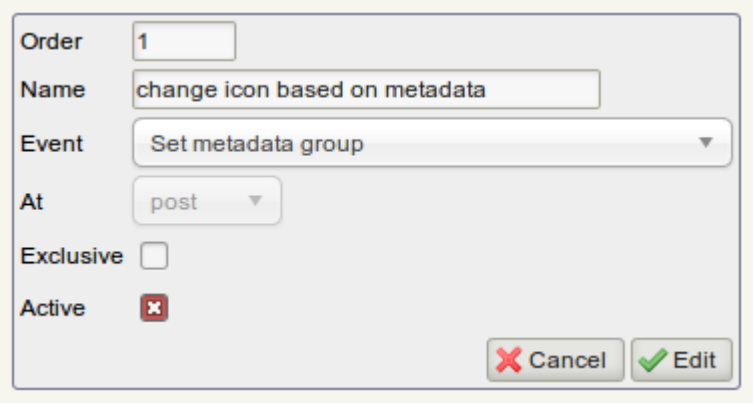
like it:

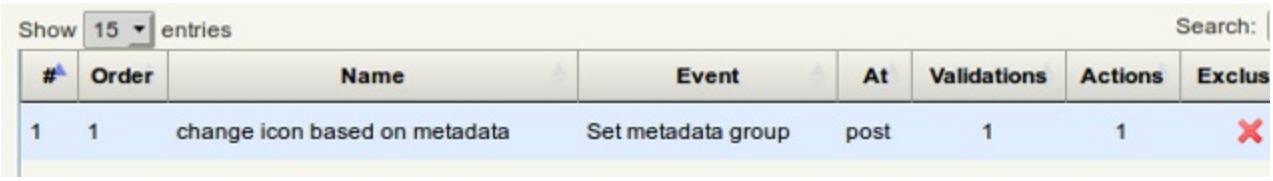
*http://localhost:8080/openkm/admin/FolderStyle?action=edit&**fs_id=2** where **the value of fs_id parameter is the value of the Id** you're looking for.*

Create a new Automation task

Create an Automation task:

- ***Set name** "Change icon based on metadata".*
- ***Choose "Set Property Group"** event (set metadata fields event).*
- ***Choose "post"** on at field.*
- ***Check Active**.*





- *Add an Action based on scripting and set the code below.*

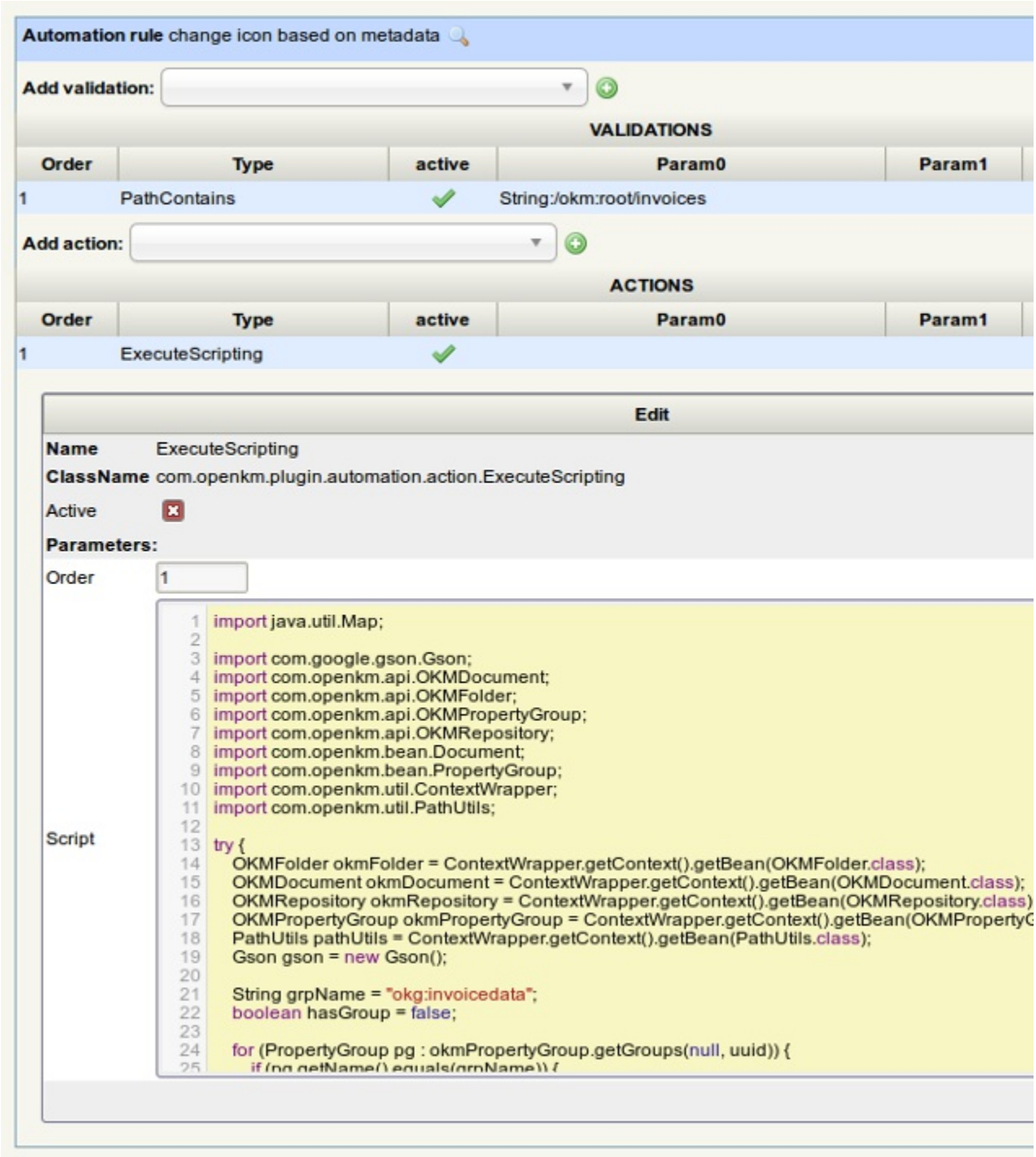
More information about Automation.

Scripting code

```
import java.util.Map;

import com.google.gson.Gson;
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.api.OKMRepository;
import com.openkm.bean.Document;
import com.openkm.bean.PropertyGroup;
import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;
```

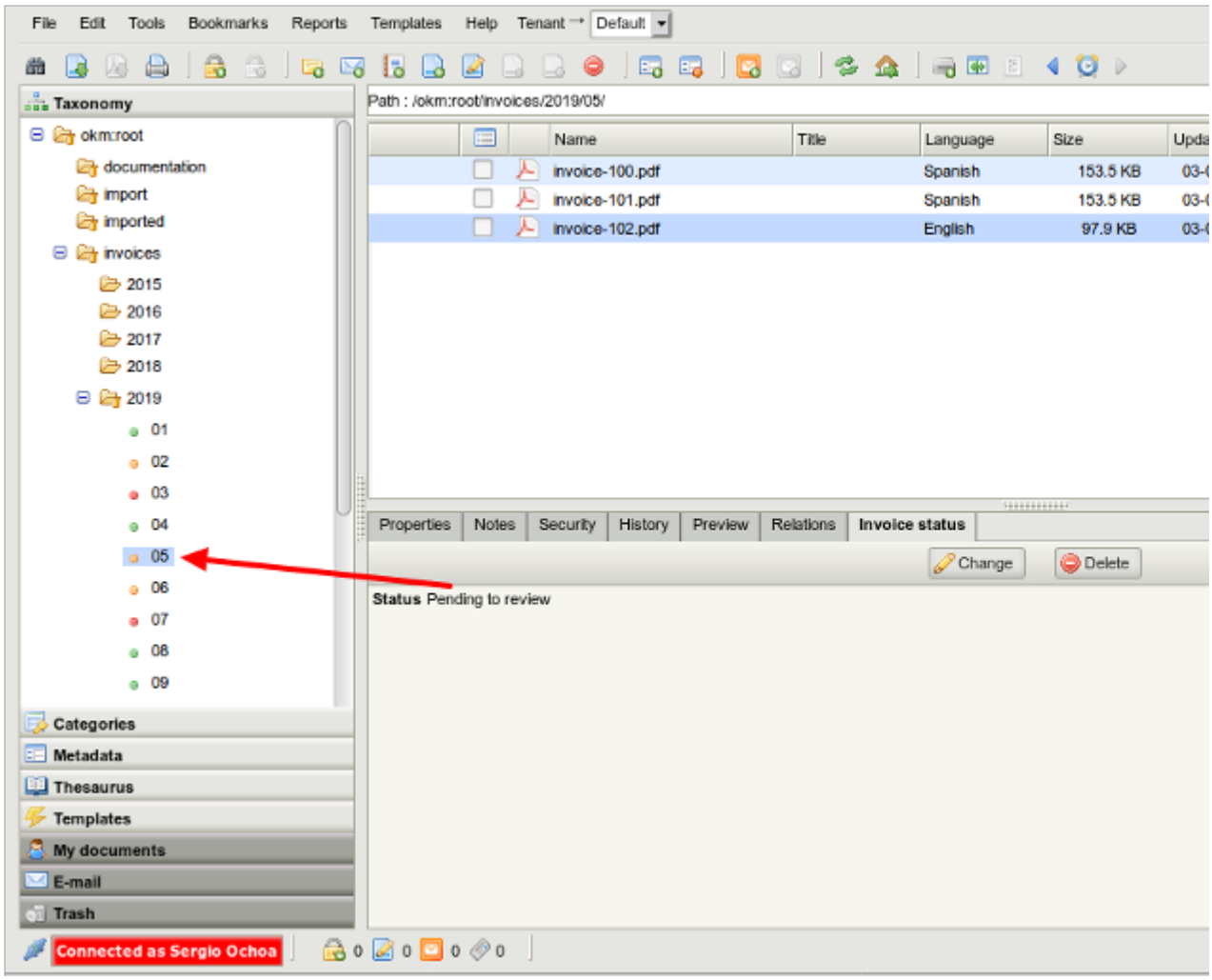
```
try {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(OKMProper
    PathUtils pathUtils = ContextWrapper.getContext().getBean(PathUtils.class);
    Gson gson = new Gson();

    String grpName = "okg:invoicedata";
    boolean hasGroup = false;

    for (PropertyGroup pg : okmPropertyGroup.getGroups(null, uuid)) {
        if (pg.getName().equals(grpName)) {
            hasGroup = true;
        }
    }

    if (hasGroup) {
        Map properties = okmPropertyGroup.getProperties(null, uuid, grpName);
        for (String key : properties.keySet()) {
            String[] values = gson.fromJson(properties.get(key), String[].class);
            String value = values[0];
            Document document = okmDocument.getProperties(null, uuid);
            String fldUuid = okmRepository.getNodeUuid(null, pathUtils.getParent(docu
            if (value.equals("open")) {
                okmFolder.setStyle(null, fldUuid, 2);
            } else if (value.equals("review")) {
                okmFolder.setStyle(null, fldUuid, 3);
            } else if (value.equals("closed")) {
                okmFolder.setStyle(null, fldUuid, 4);
            }
        }
    }
} catch (Exception e) {
    System.out.println(e.getMessage());
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

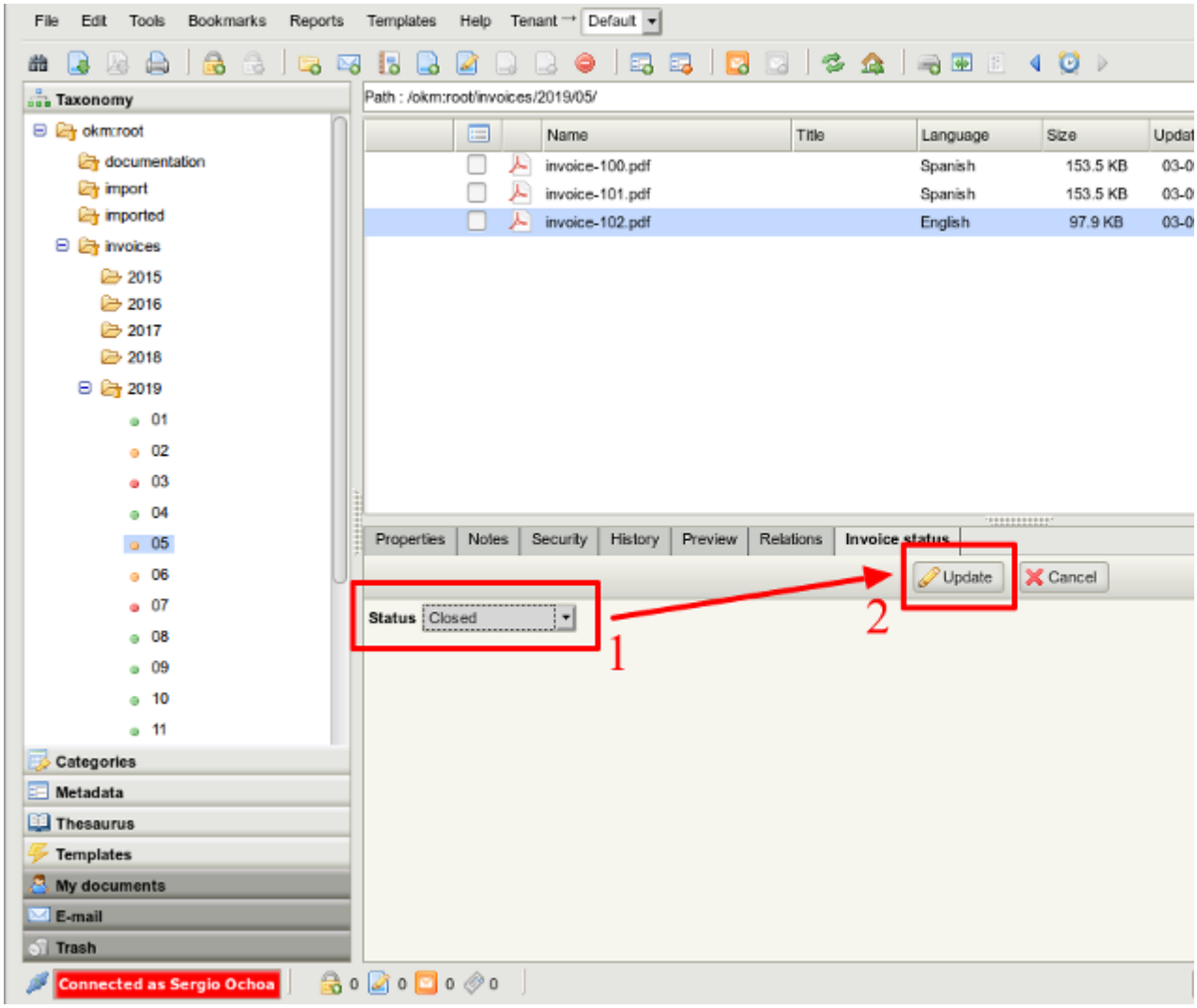


Check configuration

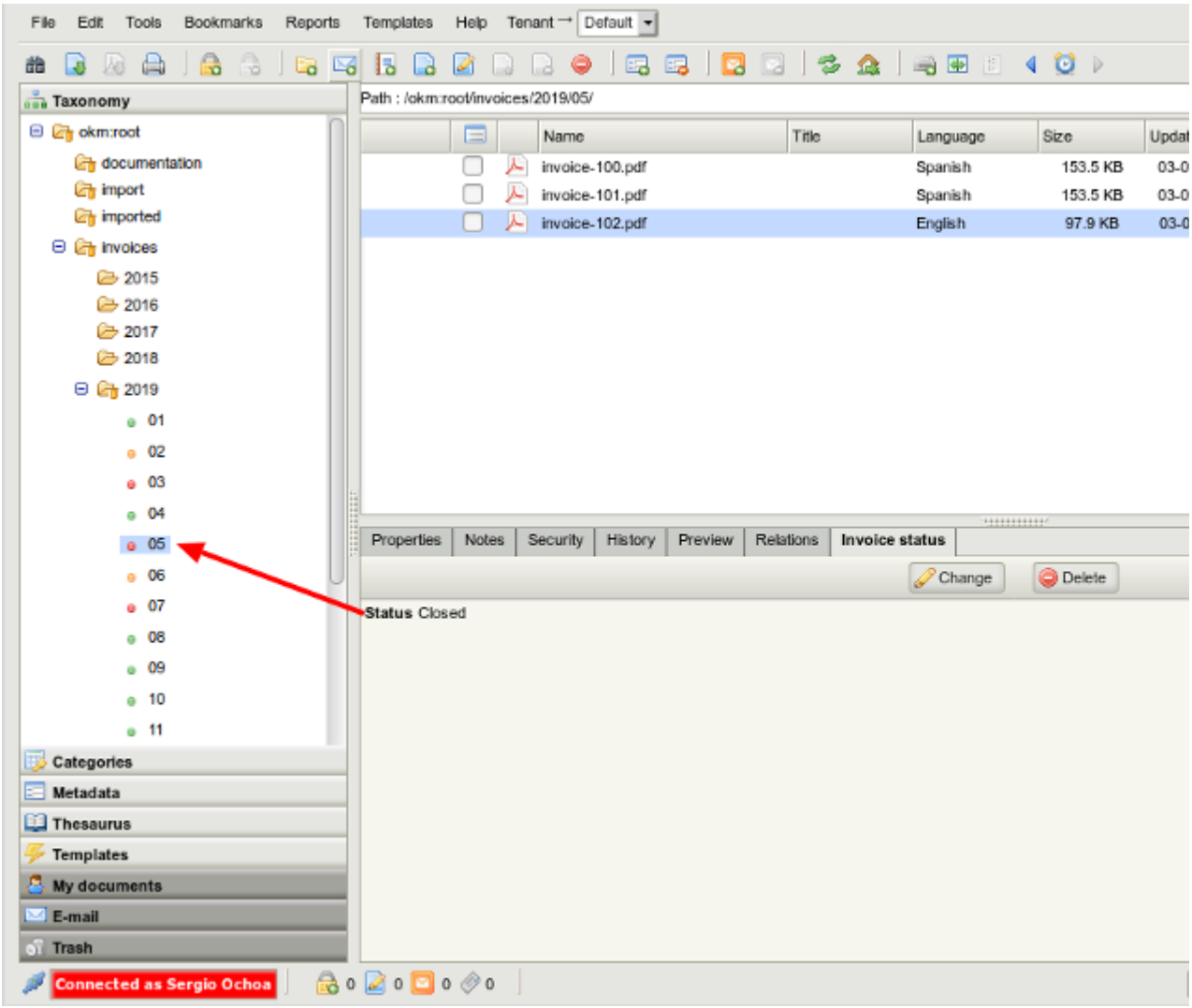
- ***Update the metadata group*** *of a document.*



- ***Change the metadata field value*** *from* ***Pending to review*** *to* ***closed***.



- *Click on* ***Refresh icon*** *at toolbar.*
- *The folder style icon applyed has been changed.*


File Edit Tools Bookmarks Reports Templates Help Tenant Default
Taxonomy
okm:root
documentation
import
imported
invoices
2015
2016
2017
2018
2019
01
02
03
04
05
06
07
08
09
10
11
Categories
Metadata
Thesaurus
Templates
My documents
E-mail
Trash
Connected as Sergio Ochoa
Path : /okm:root/invoices/2019/05/
Name
Title
Language
Size
invoice-100.pdf
Spanish
153.5 KB
invoice-101.pdf
Spanish
153.5 KB
invoice-102.pdf
English
97.9 KB
Properties
Notes
Security
History
Preview
Relations
Invoice status
Change
Delete
Status Closed

Creating a JAR file with Eclipse

These instructions are only a guide for creating a JAR based on a single .java file.

Take it as is, an introduction.

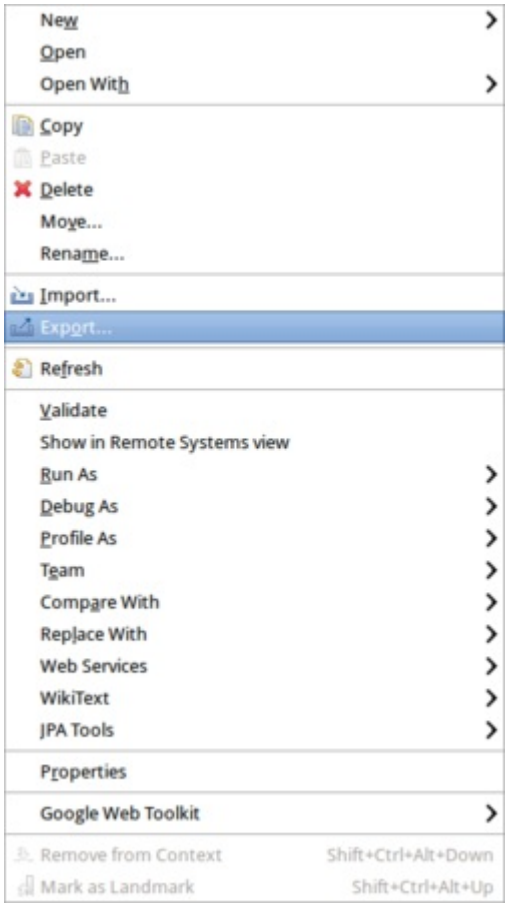
The following steps can be changed depending on the new Eclipse version changes.

1. ***Choose the Java file*** *from Project Explorer or Navigator view.*
2. ***Do right click*** *(will be shown contextual menu) and choose "export".*
3. *Choose* ***Java -> JAR file****.*
4. *Select the* ***destination folder****, for example C:\openkm-dev\jar\your-class.jar.*
5. ***Click Next****.*
6. ***Click Next****.*
7. ***Optional step****. If your java class implements "**public static void main(String[] args)**" then you must choose your java class as the class of the entry point.*
8. ***Click Finish****.*

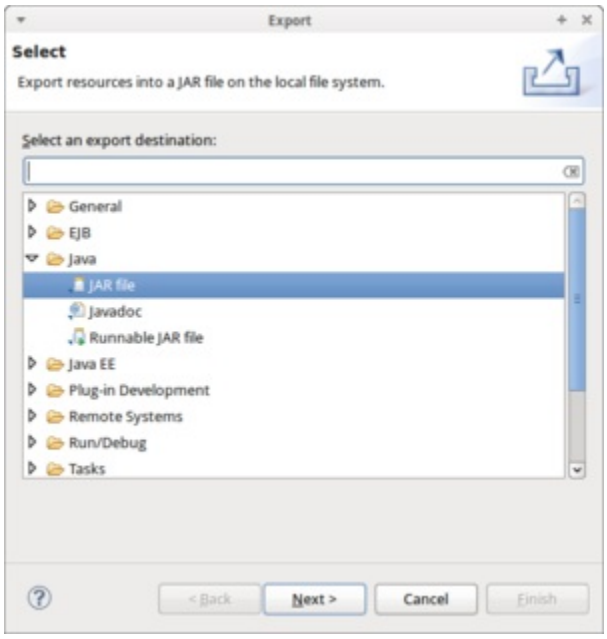
Images

Select the file to be exported as a jar. Right-click on the mouse and select " ***export****" option.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

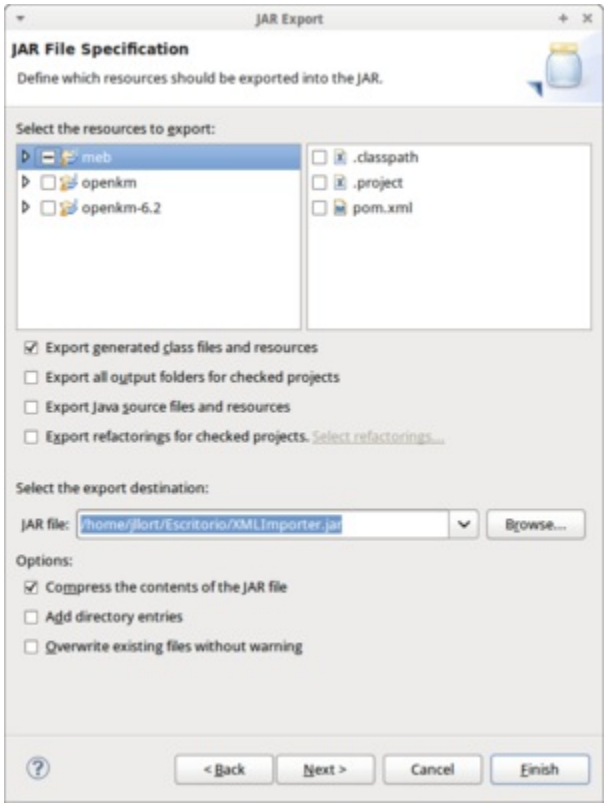


*Select "**.jar**" file option and click on "**next**" button.*

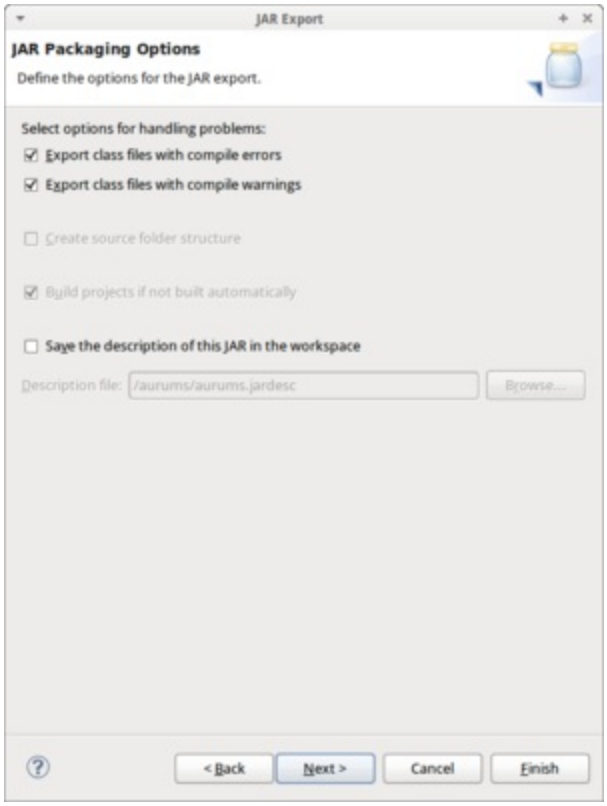


*Select the "**JAVA classes to be included**" into the jar file and click on "**next**" button.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



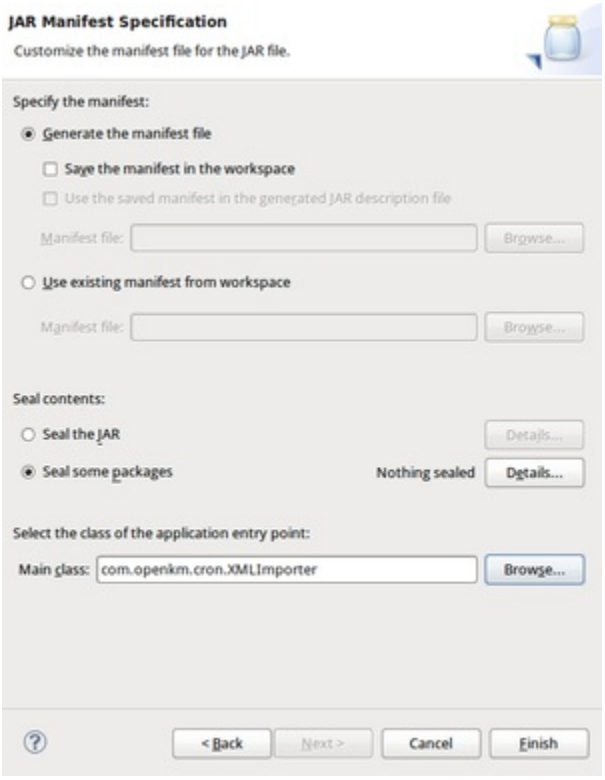
*Click on "**next**" button.*



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*Select the "**Main class**" and click on "**Finish**" button.*

> *If your JAVA class implements "**public static void main(String[] args)**" then you must choose your java Main class as the class of the entry point, otherwise you can skip this step.*



Sample JAR file

Requisites:

- *Must implement the class "**public static void main(String[] args)**".*

```
package com.openkm;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class Test {
    private static Logger log = LoggerFactory.getLogger(Test.class);

    public static void main(String[] args) {
        System.out.println("Test sample");
    }
}
```

Creating plugins

OpenKM plugin system helps you to add software components increasing the application features.

*OpenKM plugin system is based on **hot deploy** of JAR packages.*

From OpenKM plugin system you can:

- *Register new Antivirus analyzers.*
- *Register new Automation actions.*
- *Register new Automation validations.*
- *Register new form validators.*
- *Register new NodeProperties.*
- *Register new OCR validation controls (new OCR control points used to identify documents).*
- *Register new OCR data capture (new OCR parsers to retrieve data).*
- *Register new option select values (retrieve list values from third party software).*
- *Register new principal adapters (connectors to retrieve users and roles from third party software).*
- *Register new rest plugin (extend rest API).*
- *Register new suggestion analyzer (build your own logic to suggest form values).*
- *Register new shell commands (experimental feature).*
- *Register new text extrator.*
- *Register new version number adapter (create your own document version number).*
- *Register new security access manager*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Register a new plugin

- *To install new plugin **create a jar file** and **copy into your $TOMCAT/plugins** folder. More information at Creating a JAR file with Eclipse.*
- *Go to **Administration > Utilities > Plugins** and click at top right the Reload plugins button.*
- *In **plugins table will be shown** the new plugin.*

A plugin configuration needs to change Configuration parameters or restart application. For example Creating your own Principal Adapter case.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Antivirus Analyzer plugin

You can create your own Antivirus Analyzer.

Conditions:

- *The new Antivirus Analyzer class* ***must implement the "Antivirus"*** *interface.*
- *The new Antivirus Analyzer class* ***must be declared under the package "com.openkm.plugin.antivirus".***
- *The new Antivirus Analyzer class* ***must be annotated with "@PluginImplementation"****.*
- *The new Antivirus Analyzer class* ***must extend of "BasePlugin".***

Antivirus Analyzer interface:

```
package com.openkm.plugin.antivirus;

import java.io.File;

import net.xeoh.plugins.base.Plugin;

/**
 * Antivirus
 */
public interface Antivirus extends Plugin {

 /**
  * Check for viruses in file
  */
 String detect(File file);
}
```

The ***new class must be loaded into the package com.openkm.plugin.antivirus*** *because the application plugins system will try to load it from there.*

Do not miss the tag ***@PluginImplementation*** *otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Method description

Method	*Type*	*Description*
detect(File file)	*String*	*The method check the file for viruses. In case the retun value will not be null the application will consider it has been found some virus.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example of the clamav antivirus analyzer implementation

```
package com.openkm.plugin.antivirus;

import com.openkm.core.Config;
import com.openkm.plugin.BasePlugin;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.apache.commons.io.IOUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.io.File;
import java.io.IOException;

/**
 * Default implementation of an antivirus.
 *
 */
@PluginImplementation
public class ClamavAntivirus extends BasePlugin implements Antivirus {

    private static Logger log = LoggerFactory.getLogger(ClamavAntivirus.class);

    /**
     * Check for viruses in file
     */
    @Override
    public String detect(File file) {
        try {
            // Performs virus check
            log.debug("CMD: " + Config.SYSTEM_ANTIVIR + " " + file.getPath());
            ProcessBuilder pb = new ProcessBuilder(Config.SYSTEM_ANTIVIR, "--no-summa
            Process process = pb.start();
            process.waitFor();
            String info = IOUtils.toString(process.getInputStream());
            process.destroy();

            // Check return code
            if (process.exitValue() == 1) {
                log.warn(info);
                info = info.substring(info.indexOf(':') + 1);
                return info;
            } else {
                return null;
            }
        } catch (InterruptedException | IOException e) {
            log.warn("Failed to check for viruses", e);
        }

        return "Failed to check for viruses";
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Automation Action

You can create your own Automation Action.

Conditions:

- *The new Automation Action class **must implement the "Action"** interface.*
- *The new Automation Action class **must be declared under the package "com.openkm.plugin.automation.action".***
- *The new Automation Action class **must be annotated with "@PluginImplementation"**.*
- *The new Automation Action class **must extend of "BasePlugin".***

Automation Action interface:

```
package com.openkm.plugin.automation;

import com.openkm.db.bean.AutomationRule.EnumEvents;
import net.xeoh.plugins.base.Plugin;

import java.util.List;
import java.util.Map;

public interface Action extends Plugin {

    void executePre(Map<String, Object> env, Object... params) throws AutomationExcep

    void executePost(Map<String, Object> env, Object... params) throws AutomationExce

    String getName();

    String getParamType00();

    String getParamSrc00();

    String getParamDesc00();

    String getParamType01();

    String getParamSrc01();

    String getParamDesc01();

    String getParamType02();

    String getParamSrc02();

    String getParamDesc02();

    List<EnumEvents> getValidEventsAtPre();

    List<EnumEvents> getValidEventsAtPost();
}
```

*The **new class must be loaded into the package com.openkm.plugin.automation.action** because application plugins system will try to load from there.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information about Register a new plugin.

Methods description

Method	*Type*	*Description*
executePre(Map<String, Object> env, Object... params)	*void*	*The method executed by the Automation event when the validation conditions succeed, on "pre" stage.*
executePost(Map<String, Object> env, Object... params)	*void*	*The method executed by the Automation event when the validation conditions succeed, on "post" stage.*
getName()	*String*	*Sets the name that will be shown in the administrator user interface selector list.*
getParamType00 ***getParamType01*** ***getParamType02***	*String*	*Sets the parameter type.* *Available values:* • *Automation.PARAM_TYPE_EMPTY* • *Automation.PARAM_TYPE_TEXT* • *Automation.PARAM_TYPE_INTEGER* • *Automation.PARAM_TYPE_BOOLEAN* • *Automation.PARAM_TYPE_TEXTAREA* • *Automation.PARAM_TYPE_CODE* • *Automation.PARAM_TYPE_USER* • *Automation.PARAM_TYPE_ROLE* • *Automation.PARAM_TYPE_OMR* *When value is set to Automation.PARAM_TYPE_EMPTY* *The parameters work as a logical group.* ***[getParamType00 + getParamSrc00 + getParamDesc00]***

getParamSrc00 ***getParamSrc01*** ***getParamSrc02***	*String*	*Sets the source type.* *Available values:* • *Automation.PARAM_SOURCE_EMPTY* • *Automation.PARAM_SOURCE_FOLDER* • *Automation.PARAM_SOURCE_OMR*
getParamDesc00 ***getParamDesc01*** ***getParamDesc02***	*String*	*The parameter description.*
getValidEventsAtPre()	*List<EnumEvents>*	*Return a list of events.* *The method returns a list of valid events for the action in pre.*
getValidEventsAtPost()	*List<EnumEvents>*	*Return a list of events.* *The method returns a list of valid events for the action in post.*

Understanding env variable

The env variable is present on isValid method, it is a Map of values injected by automation. These map values provide information about the node involved on the event and other related information.

> *It takes some time, until you take control of the AutomationUtils class, because they are centralized, the methods to retrieve data.*
>
> *Take a look at Variables by automation event to know what variables are available for each event.*

For example to retrieve the node that caused the event.

```
NodeBase node = automationUtils.getNodeToEvaluate(env);
```

Understanding params variable

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The env variable is present on isValid method, it is an array of Objects. This array is filled by Automation event based on the number of parameters set on Action class (which will not be Automation.PARAM_TYPE_EMPTY value).

The array can have serveral Objects with a distinct type, it is necessary retrieve the parameters to the correct object type.

> *It takes some time, until you take control of AutomationUtils class, because they are centralized, the methods to retrieve parameters.*

For example to retrieve the first parameters as String object.

```
String param00 = AutomationUtils.getString(0, params);
```

Description of available ParamType values:

Property	*Description*
Automation.PARAM_TYPE_EMPTY	*Indicates an empty value.*
Automation.PARAM_TYPE_TEXT	*Indicates a text value will be required.*
Automation.PARAM_TYPE_INTEGER	*Indicates an integer value will be required.*
Automation.PARAM_TYPE_BOOLEAN	*Indicates a boolean value will be required.*
Automation.PARAM_TYPE_TEXTAREA	*Indicates a text area value will be required.*
Automation.PARAM_TYPE_CODE	*Indicates that code will be required.*
Automation.PARAM_TYPE_USER	*Indicates that a valid application user will be required.*
Automation.PARAM_TYPE_ROLE	*Indicates that a valid application role will be required.*
Automation.PARAM_TYPE_OMR	*Indicates that a valid OMR id - object mark recognition - will be required.*

Description of available Src values:

Property	*Description*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

PARAM_SOURCE_EMPTY	*Indicates an empty source.*
PARAM_SOURCE_FOLDER	*Indicates that the source must be a folder.*
PARAM_SOURCE_OMR	*Indicates that the source must be a valid OMR - object mark recognition -.*

Detecting recursion

When you create your automation actions, should consider recursion case. For example if your action goes linked with "CREATE_DOCUMENT" event and into the action you create a new document you can go into an infinite loop. For it when you create a new Action should take special care of this cases.

Basic recursive samples:

- *Automation task linked with "**Create document**" event when into is set an **action** that also **creates a new document**.*
- *Automation task linked with "**Set metadata group**" event when into is set an **action** that also **changes metadata group values**.*

Complex recursive sample:

- *Automation task linked with "**Create document**" event when into is set an action that **changes metadata group values** and automation task linked with "**Set metadata group**" event when into is set an action that **creates a new document**.*

Code to detect recursive calls caused by automation action classes:

```
if (StackTraceUtils.isCallingMe(this.getClass().getName())) {
  log.info("Recursion detected");
  return;
}
```

Example

***AddKeyword** class:*

```
package com.openkm.plugin.automation.action;

import com.openkm.core.Config;
import com.openkm.db.bean.Automation;
import com.openkm.db.bean.AutomationRule.EnumEvents;
import com.openkm.db.service.NodeBaseSrv;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.automation.Action;
```

```
import com.openkm.plugin.automation.AutomationException;
import com.openkm.plugin.automation.AutomationUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
import java.util.stream.Stream;

/**
 * AddKeyword
 */
@PluginImplementation
public class AddKeyword extends BasePlugin implements Action {

    private static ArrayList<EnumEvents> EVENTS_AT_PRE = new ArrayList<>();

    private static ArrayList<EnumEvents> EVENTS_AT_POST
            = Stream.of(EnumEvents.EVENT_DOCUMENT_CREATE, EnumEvents.EVENT_DOCUMENT_U
                    EnumEvents.EVENT_DOCUMENT_RENAME, EnumEvents.EVENT_DOCUMENT_MOVE,
                    EnumEvents.EVENT_DOCUMENT_DOWNLOAD_FROM_UI, EnumEvents.EVENT_DOCU
                    EnumEvents.EVENT_DOCUMENT_PRINT, EnumEvents.EVENT_DOCUMENT_STAMP,
                    EnumEvents.EVENT_RECORD_DELETE, EnumEvents.EVENT_FOLDER_CREATE, E
                    EnumEvents.EVENT_MAIL_CREATE, EnumEvents.EVENT_MAIL_DOWNLOAD_FROM
                    EnumEvents.EVENT_TEXT_EXTRACTOR, EnumEvents.EVENT_PROPERTY_GROUP_
                    EnumEvents.EVENT_PROPERTY_GROUP_REMOVE, EnumEvents.EVENT_MAIL_IMP
            ).collect(Collectors.toCollection(ArrayList::new));

    @Autowired
    private NodeBaseSrv nodeBaseSrv;

    @Autowired
    private AutomationUtils automationUtils;

    @Override
    public void executePre(Map<String, Object> env, Object... params) throws Automati
    }

    @Override
    public void executePost(Map<String, Object> env, Object... params) throws Automat
        try {
            String keyword = automationUtils.getString(0, params);
            String uuid = automationUtils.getNodeToEvaluate(env).getUuid();

            if (uuid != null && keyword != null && !keyword.isEmpty()) {
                if (Config.SYSTEM_KEYWORD_LOWERCASE) {
                    keyword = keyword.toLowerCase();
                }

                nodeBaseSrv.addKeyword(uuid, keyword);
            }
        } catch (Exception e) {
            throw new AutomationException("AddKeyword exception", e);
        }
    }

    @Override
    public String getName() {
        return "AddKeyword";
    }

    @Override
    public String getParamType00() {
        return Automation.PARAM_TYPE_TEXT;
    }
```

```
    @Override
    public String getParamSrc00() {
        return Automation.PARAM_SOURCE_EMPTY;
    }

    @Override
    public String getParamDesc00() {
        return "Keyword";
    }

    @Override
    public String getParamType01() {
        return Automation.PARAM_TYPE_EMPTY;
    }

    @Override
    public String getParamSrc01() {
        return Automation.PARAM_SOURCE_EMPTY;
    }

    @Override
    public String getParamDesc01() {
        return "";
    }

    @Override
    public String getParamType02() {
        return Automation.PARAM_TYPE_EMPTY;
    }

    @Override
    public String getParamSrc02() {
        return Automation.PARAM_SOURCE_EMPTY;
    }

    @Override
    public String getParamDesc02() {
        return "";
    }

    @Override
    public List<EnumEvents> getValidEventsAtPre() {
        return EVENTS_AT_PRE;
    }

    @Override
    public List<EnumEvents> getValidEventsAtPost() {
        return EVENTS_AT_POST;
    }
}
```

Creating your own Automation Validation

You can create your own Automation Validation.

Conditions:

- *The new Automation Validation class **must implement the "Validation"** interface.*
- *The new Automation Validation class **must be declared under the package "com.openkm.plugin.automation.validation".***
- *The new Automation Validation class **must be annotated with "@PluginImplementation"**.*
- *The new Automation Validation class **must extend of "BasePlugin".***

Automation Validation interface:

```
package com.openkm.plugin.automation;

import com.openkm.db.bean.AutomationRule.EnumEvents;
import net.xeoh.plugins.base.Plugin;

import java.util.List;
import java.util.Map;

public interface Validation extends Plugin {

    String METHOD = "isValid";

    boolean isValid(Map<String, Object> env, Object... params) throws AutomationExcep

    String getName();

    String getParamType00();

    String getParamSrc00();

    String getParamDesc00();

    String getParamType01();

    String getParamSrc01();

    String getParamDesc01();

    String getParamType02();

    String getParamSrc02();

    String getParamDesc02();

    List<EnumEvents> getValidEventsAtPre();

    List<EnumEvents> getValidEventsAtPost();
}
```

*The **new class must be loaded into the package com.openkm.plugin.automation.validation** because application plugins system will try to load from there.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information about Register a new plugin.

Methods description

Method	*Type*	*Description*
isValid(Map<String, Object> env, Object... params)	*boolean*	*The method executed by the Automation event to evaluate if conditions are true or false.*
getName()	*String*	*Sets the name that will be shown in the administrator user interface selector list.*
getParamType00 ***getParamType01*** ***getParamType02***	*String*	*Sets the parameter type.* *Available values:* • *Automation.PARAM_TYPE_EMPTY* • *Automation.PARAM_TYPE_TEXT* • *Automation.PARAM_TYPE_INTEGER* • *Automation.PARAM_TYPE_BOOLEAN* • *Automation.PARAM_TYPE_TEXTAREA* • *Automation.PARAM_TYPE_CODE* • *Automation.PARAM_TYPE_USER* • *Automation.PARAM_TYPE_ROLE* • *Automation.PARAM_TYPE_OMR* *When the value is set to Automation.PARAM_TYPE_EMPTY* *The parameters work as a logical group.* ***[getParamType00 + getParamSrc00 + getParamDesc00]***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getParamSrc00 ***getParamSrc01*** ***getParamSrc02***	*String*	*Set the source type.* *Available values:* • *Automation.PARAM_SOURCE_EMPTY* • *Automation.PARAM_SOURCE_FOLDER* • *Automation.PARAM_SOURCE_OMR*
getParamDesc00 ***getParamDesc00*** ***getParamDesc00***	*String*	*The parameter description.*
getValidEventsAtPre()	*List<EnumEvents>*	*Return a list of events.* *The method returns a list of valid events for the validation in pre.*
getValidEventsAtPost()	*List<EnumEvents>*	*Return a list of events.* *The method returns a list of valid events for the validation in post.*

Understanding env variable

The env variable is present on isValid method, it is a Map of values injected by automation. These map values provide information about the node involved on event and other related information.

> *It takes some time, until you take control of AutomationUtils class, because they are centralized, the methods for retrieve data.*
>
> *Take a look at Variables by automation event to know what variables are available for each event.*

For example to retrieve the node that caused the event.

```
NodeBase node = automationUtils.getNodeToEvaluate(env);
```

Understanding params variable

The env variable is present on isValid method, it is an array of Objects. This array is filled by Automation event based on the number of parameters set on Validation class (which will not be Automation.PARAM_TYPE_EMPTY value).

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The array can have serveral Objects with a distinct type, it is necessary to retrieve the parameters to the correct object type.

> *It takes some time, until you take control of AutomationUtils class, because they are centralized, the methods to retrieve parameters.*

For example to retrive the first parameters as String object.

```
String param00 = AutomationUtils.getString(0, params);
```

Description of available ParamType values:

Property	*Description*
Automation.PARAM_TYPE_EMPTY	*Indicates an empty value.*
Automation.PARAM_TYPE_TEXT	*Indicates that a text value will be required.*
Automation.PARAM_TYPE_INTEGER	*Indicates that an integer value will be required.*
Automation.PARAM_TYPE_BOOLEAN	*Indicates that a boolean value will be required.*
Automation.PARAM_TYPE_TEXTAREA	*Indicates that a text area value will be required.*
Automation.PARAM_TYPE_CODE	*Indicates that a code will be required.*
Automation.PARAM_TYPE_USER	*Indicates that a valid application user will be required.*
Automation.PARAM_TYPE_ROLE	*Indicates that a valid application role will be required.*
Automation.PARAM_TYPE_OMR	*Indicates that a valid OMR id - object mark recognition - will be required.*

Description of available Src values:

Property	*Description*
PARAM_SOURCE_EMPTY	*Indicates an empty source.*

PARAM_SOURCE_FOLDER	*Indicates that the source must be a folder.*
PARAM_SOURCE_OMR	*Indicates that the source must be a valid OMR - object mark recognition -.*

Example

PathContains *class:*

```
package com.openkm.plugin.automation.validation;

import com.openkm.api.OKMRepository;
import com.openkm.db.bean.Automation;
import com.openkm.db.bean.AutomationRule.EnumEvents;
import com.openkm.db.bean.NodeBase;
import com.openkm.db.service.NodeBaseSrv;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.automation.AutomationException;
import com.openkm.plugin.automation.AutomationUtils;
import com.openkm.plugin.automation.Validation;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
import java.util.stream.Stream;

/**
 * Check if the current parent path contains a designed one. The only parameter
 * is a path and will test if this one is included in the actual parent.
 */
@PluginImplementation
public class PathContains extends BasePlugin implements Validation {

    private static ArrayList<EnumEvents> EVENTS_AT_PRE
            = Stream.of(EnumEvents.EVENT_DOCUMENT_CREATE, EnumEvents.EVENT_DOCUMENT_U
                    EnumEvents.EVENT_DOCUMENT_RENAME, EnumEvents.EVENT_DOCUMENT_MOVE,
                    EnumEvents.EVENT_DOCUMENT_DOWNLOAD_FROM_UI, EnumEvents.EVENT_DOCU
                    EnumEvents.EVENT_DOCUMENT_PRINT, EnumEvents.EVENT_DOCUMENT_STAMP,
                    EnumEvents.EVENT_RECORD_DELETE, EnumEvents.EVENT_FOLDER_CREATE, E
                    EnumEvents.EVENT_MAIL_CREATE, EnumEvents.EVENT_MAIL_DOWNLOAD_FROM
                    EnumEvents.EVENT_TEXT_EXTRACTOR, EnumEvents.EVENT_PROPERTY_GROUP_
                    EnumEvents.EVENT_PROPERTY_GROUP_REMOVE
            ).collect(Collectors.toCollection(ArrayList::new));

    private static ArrayList<EnumEvents> EVENTS_AT_POST
            = Stream.of(EnumEvents.EVENT_DOCUMENT_CREATE, EnumEvents.EVENT_DOCUMENT_U
                    EnumEvents.EVENT_DOCUMENT_RENAME, EnumEvents.EVENT_DOCUMENT_MOVE,
                    EnumEvents.EVENT_DOCUMENT_DOWNLOAD_FROM_UI, EnumEvents.EVENT_DOCU
                    EnumEvents.EVENT_DOCUMENT_PRINT, EnumEvents.EVENT_DOCUMENT_STAMP,
                    EnumEvents.EVENT_RECORD_DELETE, EnumEvents.EVENT_FOLDER_CREATE, E
                    EnumEvents.EVENT_MAIL_CREATE, EnumEvents.EVENT_MAIL_DOWNLOAD_FROM
                    EnumEvents.EVENT_TEXT_EXTRACTOR, EnumEvents.EVENT_PROPERTY_GROUP_
                    EnumEvents.EVENT_PROPERTY_GROUP_REMOVE, EnumEvents.EVENT_MAIL_IMP
            ).collect(Collectors.toCollection(ArrayList::new));

    @Autowired
    private AutomationUtils automationUtils;
```

```
    @Autowired
    private OKMRepository okmRepository;

    @Autowired
    private NodeBaseSrv nodeBaseSrv;

    @Override
    public boolean isValid(Map<String, Object> env, Object... params) throws Automati
        try {
            String fldUuid = automationUtils.getString(0, params);
            NodeBase node = automationUtils.getNodeToEvaluate(env);
            String parentPath = nodeBaseSrv.getPathFromUuid(node.getParent());
            String fldPath = okmRepository.getNodePath(null, fldUuid);

            if (parentPath.startsWith(fldPath)) {
                return true;
            } else {
                return false;
            }
        } catch (Exception e) {
            throw new AutomationException("PathContains exception", e);
        }
    }

    @Override
    public String getName() {
        return "PathContains";
    }

    @Override
    public String getParamType00() {
        return Automation.PARAM_TYPE_TEXT;
    }

    @Override
    public String getParamSrc00() {
        return Automation.PARAM_SOURCE_FOLDER;
    }

    @Override
    public String getParamDesc00() {
        return "String";
    }

    @Override
    public String getParamType01() {
        return Automation.PARAM_TYPE_EMPTY;
    }

    @Override
    public String getParamSrc01() {
        return Automation.PARAM_SOURCE_EMPTY;
    }

    @Override
    public String getParamDesc01() {
        return "";
    }

    @Override
    public String getParamType02() {
        return Automation.PARAM_TYPE_EMPTY;
    }

    @Override
    public String getParamSrc02() {
```

```
        return Automation.PARAM_SOURCE_EMPTY;
    }

    @Override
    public String getParamDesc02() {
        return "";
    }

    @Override
    public List<EnumEvents> getValidEventsAtPre() {
        return EVENTS_AT_PRE;
    }

    @Override
    public List<EnumEvents> getValidEventsAtPost() {
        return EVENTS_AT_POST;
    }
}
```

Creating your own Crontab plugin

You can create your own Antivirus Analyzer.

Conditions:

- *The new Antivirus Analyzer class **must implement the "CronAdapter"** interface.*
- *The new Antivirus Analyzer class **must be declared under the package "com.openkm.plugin.cron".***
- *The new Antivirus Analyzer class **must be annotated with "@PluginImplementation"**.*
- *The new Antivirus Analyzer class **must extend of "BaseCronPlugin".***

Antivirus Analyzer interface:

```
package com.openkm.plugin.cron;

import net.xeoh.plugins.base.Plugin;

public interface CronAdapter extends Runnable, Plugin {
 /**
  * The cron name
  */
 String getName();

 /**
  * The default cron expression what will be registered the first time
  */
 String getCronExpression();

 /**
  * Return if task is running or not
  */
 boolean isRunning();

 /**
  * Cron task execution
  */
 void execute();
}
```

*The **new class must be loaded into the package com.openkm.plugin.cron** because the application plugins system will try to load it from there.*

*Do not miss the tag **@PluginImplementation** otherwise, the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

Method description

Method	Type	Description
getName()	String	Return the name of the crontab.
getCronExpression	String	Return the cron expression.
isRunning	Boolean	Return true when cron task is running.
execute	Void	Execute the cron task.

Example of the disposition cron implementation

```
package com.openkm.plugin.cron;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.util.FileLogger;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;


@PluginImplementation
public class TestCron extends BaseCronPlugin implements CronAdapter {
    private static Logger log = LoggerFactory.getLogger(TestCron.class);
    private static String LOG_NAME = "TestCron";

    @Autowired
    private OKMRepository okmRepository;

    @Override
    public String getName() {
        return "Cron test";
    }

    @Override
    public String getCronExpression() {
        return "0 * * * * *";
    }

    @Override
    public void execute() {
        try {
            FileLogger.info(LOG_NAME, ">>> Started");
            // Use always system token in your cron
            String token = DbSessionManager.getInstance().getSystemToken();
            Folder rootFolder = okmRepository.getRootFolder(token);
            FileLogger.info(LOG_NAME, "Root folder uuid: " + rootFolder.getUuid());
            FileLogger.info(LOG_NAME, ">>> Finished");
        } catch (Exception e) {
            log.error(e.getMessage());
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Understanding the crontab job execution context

Execution context:

- *Crontab job is executed by internal users named "system" (**the job has not a valid user session**).*
- *Crontab job is executed by the administrator user (**the job has a valid user session)** .*

> *All API methods have a variable named token.*
>
> *When **a user is logged** the call to the API use **token value to null**. That indicates the application must use the logged session.*
>
> *When **a user is not logged**, the call to the API must use a **not null token**, otherwise will get an error.*
>
> *Example of use:*
>
> - *logged user -> **valid** API call -> OKMFolder.getInstance().createSimple(null, "/okm:root/test");*
> - *not logged user -> **invalid** API call -> OKMFolder.getInstance().createSimple(**null**, "/okm:root/test");*

Use the system token is a good practice when calling API methods into crontab task:

```
String systemToken = DbSessionManager.getInstance().getSystemsystemToken();
OKMFolder.getInstance().createSimple(systemToken, "/okm:root/test")
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Default Form Values

You can create your own Default form Values.

Conditions:

- *The new Default Form Values class* ***must implement the "FormDefaultValues"*** *interface.*
- *The new Default Form Values class* ***must be declared under the package "com.openkm.plugin.form.values".***
- *The new Default Form Values class* ***must be annotated with "@PluginImplementation".***
- *The new Default Form Values class* ***must extend of "BasePlugin".***

Default Form Values interface:

```
package com.openkm.plugin.form.values;

import com.openkm.bean.form.FormElement;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface FormDefaultValues extends Plugin {

    String getName();

    void setDefaultValues(String grpName, List<FormElement> formElementList);
}
```

The ***new class must be loaded into the package com.openkm.plugin.form.values*** *because the application plugins system will try to load it from there. See the sample below:*

Do not miss the tag ***@PluginImplementation*** *otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Method description

Method	*Type*	*Description*
setDefaultValues(String grpName, List<FormElement> formElementList)	*void*	*The method sets defaults values for the form elements of the group.*

Example of the default form values implementation

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm.plugin.form.values;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;

import com.openkm.bean.form.FormElement;
import com.openkm.bean.form.Input;
import com.openkm.plugin.BasePlugin;

import net.xeoh.plugins.base.annotations.PluginImplementation;

/**
 * Implementation of default form values
 */
@PluginImplementation
public class FormDefaultValuesDate extends BasePlugin implements FormDefaultValues {

    @Override
    public String getName() {
        return "Date Default Values";
    }

    @Override
    public void setDefaultValues(String grpName, List<FormElement> formElementList) {
        if (grpName.equalsIgnoreCase("okg:test")) {
            for (FormElement element : formElementList) {
                if (element.getName().equals("okp:test.date")) {
                    Input input = (Input) element;
                    if (input.getValue().isEmpty()) {
                        DateFormat dateFormat = new SimpleDateFormat("yyyyMMddHHmmss"
                        Date date = new Date();
                        input.setValue(dateFormat.format(date));
                    }
                }
            }
        }
    }

}
```

Metadata definition

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.8//EN" "http://www
<property-groups>
   <property-group label="Test" name="okg:test" defaultValueClassName="com.openkm.plu
      <input label="Fecha" type="date" name="okp:test.date"/>
   </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Document Converter

You can create your own Document Converter.

Conditions:

- *The new Document Converter class **must implement the "Converter"** interface.*
- *The new Document Converter class **must be declared under the package "com.openkm.plugin.conversion".***
- *The new Document Converter class **must be annotated with "@PluginImplementation"**.*
- *The new Document Converter class **must extend of "BasePlugin".***

Document Converter interface:

```
package com.openkm.plugin.conversion;

import com.openkm.core.ConversionException;
import net.xeoh.plugins.base.Plugin;

import java.io.File;
import java.io.IOException;
import java.util.List;

public interface Converter extends Plugin {

    int TO_PDF = 0;
    int TO_IMAGE = 1;

    void makeConversion(File input, String mimeType, File output) throws IOException,

    List<String> getSourceMimeTypes();

    int getConversionType();
}
```

*The **new class must be loaded into the package com.openkm.plugin.conversion** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Methods description

Method	*Type*	*Description*
makeConversion(File input, String mimeType, File output)	*void*	*The method converts the file at input to some specific format into file output*

getSourceMimeTypes()	*List<String>*	*Return a list of valid document MIME TYPE allowed to be converted.*
getConversionType()	*int*	*Return the value of the MIME TYPE of that will be converted the input document.* *Allowed values are:* • *Converter.TO_PDF* • *Converter.TO_IMAGE* *When you choose Converter.TO_IMAGE is expected to retun an image in PNG format.*

Example of the Document converter implementation

```
package com.openkm.plugin.conversion;

import com.openkm.core.ConversionException;
import com.openkm.core.MimeTypeConfig;
import com.openkm.plugin.BasePlugin;
import com.openkm.util.DocConverter;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

import java.io.File;
import java.io.IOException;
import java.util.Arrays;
import java.util.List;


@PluginImplementation
public class SqlToPdfConversion extends BasePlugin implements Converter {

    String[] sourceMimeTypes = new String[]{MimeTypeConfig.MIME_SQL};

    @Autowired
    private DocConverter docConverter;

    @Override
    public void makeConversion(File input, String mimeType, File output) throws IOExc
        docConverter.src2pdf(input, output, "sql");
    }

    @Override
    public List<String> getSourceMimeTypes() {
        return Arrays.asList(sourceMimeTypes);
    }

    @Override
    public int getConversionType() {
        return Converter.TO_PDF;
```

```
        }
    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Form Validator plugin

Form validator are used to implement complex logic that must be evaluated from server side logic and are not covered by default Metadata Validator element.

You can create your own Form Validator

Conditions:

- *The new Form Validator class **must implement the "FieldValidator"** interface.*
- *The new Form Validator class **must be declared under the package "com.openkm.plugin.form.validator".***
- *The new Form Validator class **must be annotated with "@PluginImplementation".***
- *The new Form Validator class **must extend of "BasePlugin".***

Form Validator interface:

```
package com.openkm.plugin.form;

import net.xeoh.plugins.base.Plugin;

import java.util.List;

/**
 * FieldValidator
 */
public interface FieldValidator extends Plugin {

    String getName();

    String validate(String value, List<String> uuids);
}
```

*The **new class must be loaded into the package com.openkm.plugin.form.validator** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

Methods description

Method	*Type*	*Description*
getName()	*String*	*The method returns the name of the plugin used in form list selector.*

validate(String value, List<String> uuids)	*String*	*When returned string is equals to "", application consider all goes right, otherwise text returned is the error.* *Value variable, contains the value of the metadata field and the uuids are the list of the uuid nodes affected (documents, folders, mails, records).* *In case will be updated only one node, common case, the list will only contains one uuid, in case multiple nodes selected will contains the list of all selected nodes where want to be applied the metadata changes.*

Example of the Form validator implementation

In this example it will not be allowed two equal comments in the metadata field named okp:tecnology.comment

```
package com.openkm.plugin.form.validator;

import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import com.openkm.api.OKMRelation;
import com.openkm.bean.Relation;
import com.openkm.db.service.LegacySrv;
import com.openkm.plugin.BasePlugin;

import net.xeoh.plugins.base.annotations.PluginImplementation;

@PluginImplementation
public class DuplicateDocumentNumberValidator extends BasePlugin implements FieldVali

    @Autowired
    private LegacySrv legacySrv;

    @Autowired
    private OKMRelation okmRelation;

    @Override
    public String getName() {
        return "Duplicated document number";
    }

    @Override
    public String validate(String value, List<String> uuids) {
        String validate = "";

        String sql = "SELECT NPG_NODE FROM OKM_NODE_PROPERTY WHERE NPG_NAME = 'okp:te
        try {
            List<List<String>> result = legacySrv.executeSQL(sql);
            if (result.size() > 0) {
                if (uuids.isEmpty()) {
                    validate = "Duplicated document number";
                } else {
                    List<String> allowedUuids = new ArrayList<>();
                    for (String uuid : uuids) {
                        allowedUuids.add(uuid);
                        List<Relation> relations = okmRelation.getRelations(null, uui
```

```
                        for (Relation relation : relations) {
                            allowedUuids.add(relation.getNode());
                        }
                    }

                    for (List<String> resultList : result) {
                        if (!allowedUuids.contains(resultList.get(0))) {
                            validate = "Duplicated document number";
                        }
                    }
                }
            }
        } catch (Exception e) {
            validate = e.getMessage();
        }

        return validate;
    }
}
```

Creating your own NodePaginator plugin

NodePaginator is used to customize the file browser columns shown, including metadata columns.

You can create your own NodePaginator.

Conditions:

- *The new NodePaginator class **must implement the "NodePaginator"** interface.*
- *The new NodePaginator class **must be declared under the package "com.openkm.plugin.paginate".***
- *The new NodePaginator class **must be annotated with "@PluginImplementation".***
- *The new NodePaginator class **must extend of "BasePlugin".***

NodePaginator interface:

```
package com.openkm.plugin.paginate;

import com.openkm.bean.NodePaginatorConfig;
import com.openkm.bean.PageInfo;
import com.openkm.core.AccessDeniedException;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.db.bean.NodeBase;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

/**
 * Node Paginator
 */
public interface NodePaginator extends Plugin {

 PageInfo getChildrenPaginated(String uuid, int offset, int limit, String filter, Str
 List<Class<? extends NodeBase>> filteredByNodeTypeList) throws AccessDeniedException
 DatabaseException;

 NodePaginatorConfig getConfig() throws DatabaseException;

 PageInfo getChildrenByCategoryPaginated(String uuid, int offset, int limit, String f
 boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList) throws Acc
 PathNotFoundException, DatabaseException;

 String getName();
}
```

*The **new class must be loaded into the package com.openkm.plugin.paginate** because the application plugins system will try to load it from there.*

*Do not miss the tag **@PluginImplementation** otherwise, the application plugin system will not be able to retrieve the new class.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

More information about Register a new plugin.

Method description

Method	Type	Description
getChildrenPaginated(String uuid, int offset, int limit, String filter, String orderByField, boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList)	***PageInfo***	*Get a paginated list of nodes from a parent (uuid). The object PageInfo contains the list of values for each column including metadata columns.*
getChildrenByCategoryPaginated(String uuid, int offset, int limit, String filter, String orderByField, boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList)	***PageInfo***	*Get a paginated list of nodes by category from a parent (uuid). The object PageInfo contains the list of values for each column including metadata columns.*
getConfig()	***NodePaginatorConfig***	*Used to set visible columns and their names in case of metadata columns.* *When the label of a column is not empty the columns are considered visible.*
getName()	***String***	*The name of the plugin.*

Example

Metadata XML definition:

More information at Metadata.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
<property-group label="Consulting" name="okg:consulting">
  <input label="Input label 1" name="okp:consulting.input1" dbColumnSize="256"/>
  <input label="Input label 2" name="okp:consulting.input2" dbColumnSize="512"/>
</property-group>
```

TestNodePaginator *class:*

```
package com.openkm.plugin.paginate;

import com.openkm.bean.NodePaginatorConfig;
import com.openkm.bean.PageInfo;
import com.openkm.core.AccessDeniedException;
import com.openkm.core.Config;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.db.bean.NodeBase;
import com.openkm.db.bean.Profile;
import com.openkm.db.service.LegacySrv;
import com.openkm.db.service.NodeBaseSrv;
import com.openkm.db.service.ProfileSrv;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalUtils;
import com.openkm.ws.rest.util.SimpleNodeBase;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@PluginImplementation
public class TestNodePaginator extends BasePlugin implements NodePaginator {
 private static final Logger log = LoggerFactory.getLogger(TestNodePaginator.class);
 private static final String PGRP_COLUMN_LABEL_1 = "Input 1";
 private static final String PGRP_COLUMN_LABEL_2 = "Input 2";

 @Autowired
 private NodeBaseSrv nodeBaseSrv;

 @Autowired
 private ProfileSrv profileSrv;

 @Autowired
 private LegacySrv legacySrv;

 @Override
 public PageInfo getChildrenPaginated(String uuid, int offset, int limit, String filt
 boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList) throws Acc
 PathNotFoundException, DatabaseException {
     log.debug("getChildrenPaginated({}, {}, {}, {}, {}, {}, {})", uuid, offset, limi
     PageInfo pageInfo = nodeBaseSrv.getCommonChildrenPaginated(uuid, offset, limit,

     // Only fill the metadata values when results size greater than 0
     if (pageInfo.getObjects().size() > 0) {
         setMetaDataColumns((List<SimpleNodeBase>) pageInfo.getObjects());
     }

     return pageInfo;
```

```
}

@Override
public PageInfo getChildrenByCategoryPaginated(String uuid, int offset, int limit, S
boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList) throws Acc
PathNotFoundException, DatabaseException {
    PageInfo pageInfo = nodeBaseSrv.getCommonChildrenByCategoryPaginated(uuid, offse

    // Only fill the metadata values when results size greater than 0
    if (pageInfo.getObjects().size() > 0) {
        setMetaDataColumns((List<SimpleNodeBase>) pageInfo.getObjects());
    }

    return pageInfo;
}

@Override
public String getName() {
    return "Consulting metadata sample";
}

private void setMetaDataColumns(List<SimpleNodeBase> nodes) throws DatabaseException
    Map<String, List<String>> rowValues = new HashMap<>();
    StringBuilder query = new StringBuilder("SELECT RGT_UUID, RGT_PRO_INPUT1, RGT_PR
    + "FROM OKM_PGRP_CUR_CONSULTING WHERE RGT_UUID in(");

    // Create inFilter
    StringBuilder inFilter = new StringBuilder();
    for (SimpleNodeBase snb : nodes) {
        if (inFilter.length() > 0) {
            inFilter.append(",");
        }
        inFilter.append("'");
        inFilter.append(snb.getUuid());
        inFilter.append("'");
    }
    query.append(inFilter);
    query.append(");");

    // Convert list of values to map of list
    for (List<String> row : legacySrv.executeSQL(query.toString())) {
        List<String> cols = new ArrayList<>();

        String uuid = row.get(0);
        for (int i = 1; i < row.size(); i++) {
            cols.add(row.get(i));
        }

        rowValues.put(uuid, cols);
    }

    // Set metadata field values
    for (SimpleNodeBase node : nodes) {
        if (rowValues.get(node.getUuid()) != null) {
            node.setPgrpColumnValue1(rowValues.get(node.getUuid()).get(0));
            node.setPgrpColumnValue2(rowValues.get(node.getUuid()).get(1));
        }
    }
}

public NodePaginatorConfig getConfig() throws DatabaseException {
    NodePaginatorConfig config = new NodePaginatorConfig();
    config.setPgrpColumnLabel1(PGRP_COLUMN_LABEL_1);
    config.setPgrpColumnLabel2(PGRP_COLUMN_LABEL_2);

    // Use current profile configuration for other parameters except in case of syst
    String user = PrincipalUtils.getUser();
```

```
        if (!Config.SYSTEM_USER.equals(user)) {
            Profile profile = profileSrv.findByUser(user);
            config.setColumnMassiveVisible(profile.getPrfFileBrowser().getMassiveVisible
            config.setColumnIconVisible(profile.getPrfFileBrowser().getIconVisible());
            config.setColumnNameVisible(profile.getPrfFileBrowser().getNameVisible());
            config.setColumnTitleVisible(profile.getPrfFileBrowser().getTitleVisible());
            config.setColumnLanguageVisible(profile.getPrfFileBrowser().getLanguageVisib
            config.setColumnSizeVisible(profile.getPrfFileBrowser().getSizeVisible());
            config.setColumnLastModifiedVisible(profile.getPrfFileBrowser().getLastModif
            config.setColumnAuthorVisible(profile.getPrfFileBrowser().getAuthorVisible()
            config.setColumnVersionVisible(profile.getPrfFileBrowser().getVersionVisible
        }

        return config;
    }
}
```

Creating your own NodeProperties plugin (customize search results objects)

NodeProperties is usually used from Webservices scenario. Using them you can customize the results objects from a query search. Usually as a result of some search query you only need to use few variables of the returned Objects rather than complete ones. With large query results, returning few data you get best perfomance (less time for filling data at server side and less time for marshalling and unmarshalling webservices procedure).

You can create your own NodeProperties.

Conditions:

- *The new NodeProperties class* ***must implement the "NodeProperties"*** *interface.*
- *The new NodeProperties class* ***must be declared under the package "com.openkm.plugin.properties".***
- *The new NodeProperties class* ***must be annotated with "@PluginImplementation"****.*
- *The new NodeProperties class* ***must extend of "BasePlugin".***

NodeProperties interface:

```
package com.openkm.plugin.properties;

import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.bean.Record;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.core.RepositoryException;
import com.openkm.db.bean.NodeDocument;
import com.openkm.db.bean.NodeFolder;
import com.openkm.db.bean.NodeMail;
import com.openkm.db.bean.NodeRecord;
import net.xeoh.plugins.base.Plugin;

/**
 * NodeProperties
 */
public interface NodeProperties extends Plugin {

    /**
     * Get document properties
     */
    Document getProperties(String user, NodeDocument nDocument, String linkTarget) th

    /**
     * Get folder properties
     */
    Folder getProperties(String user, NodeFolder nFolder) throws PathNotFoundExceptio

    /**
     * Get mail properties
     */
    Mail getProperties(String user, NodeMail nMail) throws PathNotFoundException, Dat

    /**
     * Get record properties
```

```
     */
    Record getProperties(String user, NodeRecord nRecord) throws PathNotFoundExceptio
}
```

*The **new class must be loaded into the package com.openkm.plugin.properties** because the application plugins system will try to load it from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Method description

Method	Type	Description
getProperties(String user, NodeDocument nDocument, String linkTarget)	***Document***	*Define which values are loaded in Document Object returned by search engine.*
getProperties(String user, NodeFolder nFolder)	***Return***	*Define which values are loaded in Folder Object returned by search engine.*
getProperties(String user, NodeMail nMail)	***Mail***	*Define which values are loaded in Mail Object returned by search engine.*
getProperties(String user, NodeRecord nRecord)	***Record***	*Define which values are loaded in Record Object returned by search engine.*

Example of NodeProperties implementation

```
package com.openkm.plugin.properties;

import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.bean.Record;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.core.RepositoryException;
import com.openkm.db.bean.NodeDocument;
import com.openkm.db.bean.NodeFolder;
import com.openkm.db.bean.NodeMail;
import com.openkm.db.bean.NodeRecord;
import com.openkm.plugin.BasePlugin;
```

```
import net.xeoh.plugins.base.annotations.PluginImplementation;

/**
 * UuidNodeProperties
 */
@PluginImplementation
public class UuidNodeProperties extends BasePlugin implements NodeProperties {

    /**
     * Get document properties
     */
    @Override
    public Document getProperties(String user, NodeDocument nDocument, String linkTar
        Document node = new Document();
        node.setUuid(nDocument.getUuid());
        return node;
    }

    /**
     * Get folder properties
     */
    @Override
    public Folder getProperties(String user, NodeFolder nFolder) throws PathNotFoundE
        Folder node = new Folder();
        node.setUuid(nFolder.getUuid());
        return node;
    }

    /**
     * Get mail properties
     */
    @Override
    public Mail getProperties(String user, NodeMail nMail) throws PathNotFoundExcepti
        Mail node = new Mail();
        node.setUuid(nMail.getUuid());
        return node;
    }

    /**
     * Get record properties
     */
    @Override
    public Record getProperties(String user, NodeRecord nRecord) throws PathNotFoundE
        Record node = new Record();
        node.setUuid(nRecord.getUuid());
        return node;
    }

}
```

Creating your own OCR Validation Control plugin

You can create your own OCR Validation plugin.

Conditions:

- *The new OCR Validation Control class **must implement the "OCRTemplateControlParser"** interface.*
- *The new OCR Validation Control class **must be declared under the package "com.openkm.plugin.ocr.template.controlparser".***
- *The new OCR Validation Control class **must be annotated with "@PluginImplementation"**.*
- *The new OCR Validation Control class **must extend of "BasePlugin".***

OCR Validation Control interface:

```
package com.openkm.plugin.ocr.template;

import com.openkm.db.bean.OCRTemplateControlField;
import net.xeoh.plugins.base.Plugin;

/**
 * OCRTemplateControlParser
 */
public interface OCRTemplateControlParser extends Plugin {

    boolean parse(OCRTemplateControlField otcf, String text) throws OCRTemplateExcept

    String getName();

    boolean isPatternRequired();

    String info();
}
```

*The **new class must be loaded into the package com.openkm.plugin.ocr.template.controlparser** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Methods description

Method	*Type*	*Description*
parse(OCRTemplateControlField	*boolean*	*When returns true indicates that the field complies with the conditions.*

otcf, String text)		*More information about OCR Template stages at: OCR templates.*
getName()	*String*	*Sets the name that will be shown in the administrator user interface selector list.*
isPatternRequired()	*boolean*	*When returns true indicates that the pattern field is required.* *It will show an error - missing pattern - on the administrator user interface if a pattern is required and the user is not set.*
info()	*String*	*Description of the implemented logic.* *This description is shown in the administrator user interface.*

Example:

StringSimilarParser *class:*

```
package com.openkm.plugin.ocr.template.controlparser;

import com.openkm.core.DatabaseException;
import com.openkm.db.bean.OCRTemplateControlField;
import com.openkm.db.service.ConfigSrv;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.ocr.template.OCRParserEmptyValueException;
import com.openkm.plugin.ocr.template.OCRTemplateControlParser;
import com.openkm.plugin.ocr.template.OCRTemplateException;
import com.openkm.util.AlgorithmUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * StringParser
 */
@PluginImplementation
public class StringSimilarParser extends BasePlugin implements OCRTemplateControlPars

    @Autowired
    private ConfigSrv configSrv;

    @Override
    public boolean parse(OCRTemplateControlField otf, String text) throws OCRTemplate
        if (text == null || text.equals("")) {
            return false;
```

```
        } else {
            if (otf.getPattern() == null || otf.getPattern().equals("")) {
                int distance = AlgorithmUtils.computeLevenshteinDistance(text, otf.ge
                try {
                    return distance < configSrv.getInteger("ocr.control.similar.dista
                } catch (DatabaseException e) {
                    throw new OCRTemplateException(e);
                }
            } else {
                Pattern pattern = Pattern.compile("(" + otf.getPattern() + ")", Patte
                Matcher matcher = pattern.matcher(text);
                if (matcher.find() && matcher.groupCount() == 1) {
                    if (otf.getValue().equals("")) {
                        return true;
                    } else {
                        int distance = AlgorithmUtils.computeLevenshteinDistance(matc
                        try {
                            return distance < configSrv.getInteger("ocr.control.simil
                        } catch (DatabaseException e) {
                            throw new OCRTemplateException(e);
                        }
                    }
                } else {
                    return false;
                }
            }
        }
    }

    @Override
    public String getName() {
        return "Similar String";
    }

    @Override
    public boolean isPatternRequired() {
        return false;
    }

    @Override
    public String info() {
        StringBuilder sb = new StringBuilder();
        sb.append("Empty pattern case: true if extracted text similar field value.\n"
        sb.append("Pattern with value field empty: true if pattern found.\n");
        sb.append("Pattern with value field: true if pattern found similar value.\n")
        return null;
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own OCR Data Capture plugin

You can create your own OCR Data Capture plugin.

Conditions:

- *The new OCR Data Capture class **must implement the "OCRTemplateParser"** interface.*
- *The new OCR Data Capture class **must be declared under the package "com.openkm.plugin.ocr.template.parser".***
- *The new OCR Data Capture class **must be annotated with "@PluginImplementation"**.*
- *The new OCR Data Capture class **must extend of "BasePlugin".***

OCR Data Capture interface:

```
package com.openkm.plugin.ocr.template;

import com.openkm.db.bean.OCRTemplateField;
import net.xeoh.plugins.base.Plugin;

/**
 * OCRTemplateParser
 */
public interface OCRTemplateParser extends Plugin {

    Object parse(OCRTemplateField otf, String text) throws OCRTemplateException, OCRP

    String getName();

    boolean isPatternRequired();
}
```

*The **new class must be loaded into the package com.openkm.plugin.ocr.template.parser** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at [Register a new plugin](#).

Methods description

Method	*Type*	*Description*
parse(OCRTemplateField otf, String text)	*Object*	*Returns the value of the data captured into an Object.* *Allowed Object classes:* • *String*

		• *Calendar (will be converted to ISO8601 String)* • *Other (will be converted to string with String.valueOf(object).*
getName()	*String*	*Sets the name that will be shown in the administrator user interface selector list.*
isPatternRequired()	*boolean*	*When returns true indicates that the pattern field is required.* *Will show an error - missing pattern - on the administrator user interface if pattern is required and the user is not set.*

Example

StringParser *class:*

```
package com.openkm.plugin.ocr.template.parser;

import com.openkm.db.bean.OCRTemplateField;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.ocr.template.OCRParserEmptyValueException;
import com.openkm.plugin.ocr.template.OCRTemplateException;
import com.openkm.plugin.ocr.template.OCRTemplateParser;
import net.xeoh.plugins.base.annotations.PluginImplementation;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * StringParser
 */
@PluginImplementation
public class StringParser extends BasePlugin implements OCRTemplateParser {

    @Override
    public Object parse(OCRTemplateField otf, String text) throws OCRTemplateExceptio
        if (text == null || text.equals("")) {
            throw new OCRParserEmptyValueException("Empty value");
        }

        if (otf.getPattern() == null || otf.getPattern().equals("")) {
            return text != null ? text.trim() : null;
        } else {
            Pattern pattern = Pattern.compile("(" + otf.getPattern() + ")", Pattern.U
            Matcher matcher = pattern.matcher(text);

            if (matcher.find() && matcher.groupCount() == 1) {
                return matcher.group();
            } else {
                throw new OCRTemplateException("Bad format, parse exception");
            }
        }
    }

    @Override
    public String getName() {
```

```
        return "String";
    }

    @Override
    public boolean isPatternRequired() {
        return false;
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Option Select Values plugin

You can create your own Option Select Values.

Conditions:

- *The new Option Select Values class* ***must implement the "OptionSelectValues"*** *interface.*
- *The new Option Select Values class* ***must be declared under the package "com.openkm.plugin.form.values".***
- *The new Option Select Values class* ***must be annotated with "@PluginImplementation".***
- *The new Option Select Values class* ***must extend of "BasePlugin".***

Option Select Values interface:

```
package com.openkm.plugin.form.values;

import com.openkm.bean.form.Option;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

/**
 * OptionSelectValues
 */
public interface OptionSelectValues extends Plugin {

    String getName();

    List<Option> getOptions() throws OptionSelectException;
}
```

The ***new class must be loaded into the package com.openkm.plugin.form.values*** *because application plugin system will try to load from there.*

Do not miss the tag ***@PluginImplementation*** *otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Methods description

Method	*Type*	*Description*
getName()	*String*	*Return the name what will be shown at plugins table.*
getOptions()	*List<Option>*	*Return a list with the options.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example of the Option select values implementation

```
package com.openkm.plugin.form.values;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.bean.form.Option;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalAdapterException;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.List;

@PluginImplementation
public class OptionSelectUserList extends BasePlugin implements OptionSelectValues {

    @Autowired
    private OKMAuth okmAuth;

    @Override
    public String getName() {
        return "User list";
    }

    @Override
    public List<Option> getOptions() throws OptionSelectException {
        List<Option> options = new ArrayList<>();

        try {
            List<CommonUser> userList = okmAuth.getUsers(null);
            for (CommonUser user : userList) {
                Option option = new Option();
                option.setLabel(user.getName());
                option.setValue(user.getId());
                options.add(option);
            }
        } catch (PrincipalAdapterException e) {
            throw new OptionSelectException(e);
        }

        return options;
    }
}
```

Creating your own Principal Adapter

Principal adapters are used by authentication module to retrieve users and roles information. For example retrieve users and roles from some LDAP, or external database that stores all the company users and roles.

You can create your own Principal Adapter

Conditions:

- *The new Principal Adapter class **must implement the "PrincipalAdapter"** interface.*
- *The new Principal Adapter class **must be declared under the package "com.openkm.plugin.principal".***
- *The new Principal Adapter class **must be annotated with "@PluginImplementation".***
- *The new Principal Adapter class **must extend of "BasePlugin".***

Principal Adapter interface:

```
package com.openkm.plugin.principal;

import com.openkm.bean.CommonUser;
import com.openkm.db.bean.Profile;
import com.openkm.principal.PrincipalAdapterException;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface PrincipalAdapter extends Plugin {

    List<CommonUser> getUsers() throws PrincipalAdapterException;

    List<String> getRoles() throws PrincipalAdapterException;

    CommonUser getUser(String userId) throws PrincipalAdapterException;

    List<CommonUser> getUsersByRole(String role) throws PrincipalAdapterException;

    List<String> getRolesByUser(String user) throws PrincipalAdapterException;

    String getPassword(String user) throws PrincipalAdapterException;

    /*
     * ------------------------------------------------------------------
     * These methods only works if using the OpenKM user database.
     * ------------------------------------------------------------------
     */

    CommonUser createUser(CommonUser user) throws PrincipalAdapterException;

    void deleteUser(String user) throws PrincipalAdapterException;

    CommonUser updateUser(CommonUser user) throws PrincipalAdapterException;

    void createRole(String role, boolean active) throws PrincipalAdapterException;

    void deleteRole(String role) throws PrincipalAdapterException;

    void updateRole(String role, boolean active) throws PrincipalAdapterException;
```

```
    void assignRole(String user, String role) throws PrincipalAdapterException;
    void removeRole(String user, String role) throws PrincipalAdapterException;
    List<Profile> getProfiles(boolean filterByActive) throws PrincipalAdapterExceptio
    Profile getUserProfile(String userId) throws PrincipalAdapterException;
    void setUserProfile(String userId, long profileId) throws PrincipalAdapterExcepti
    boolean isManageUsers();
    boolean isManageRoles();
}
```

*The **new class must be loaded into the package com.openkm.plugin.principal** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise, the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

*To **enable the new Principal adapter** go to **Administration > Configuration parameters** > find the parameter named **principal.adapter** and modify the value with **com.openkm.plugin.adapter.OwnPrincipalAdapter***

*To take effect this change we need to **restart the application**.*

Methods description

Method	*Type*	*Description*
getUsers()	*List<CommonUser>*	*Return the list of all the users.*
getRoles()	*List<String>*	*Return the list of all the roles.*
getUser(String userId)	*CommonUser*	*Return all user data*
getUsersByRole(String role)	*List<CommonUser>*	*Return the list of all the users who have assigned a role.*
getRolesByUser(String user)	*List<String>*	*Return the list of all the roles assigned to a user.*
String getPassword(String user)	*String*	*Return the password associated to a specific user.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

createUser(CommonUser user)	*void*	*Create a new user.*
deleteUser(String user)	*void*	*Delete a user.*
updateUser(CommonUser user)	*void*	*Update a user.* *Password parameter can be null or empty.*
createRole(String role, boolean active)	*void*	*Create a new role.*
deleteRole(String role)	*void*	*Delete a role.*
updateRole(String role, boolean active)	*void*	*Update a role.*
assignRole(String user, String role)	*void*	*Grant user with some role.*
removeRole(String user, String role)	*void*	*Revoke a role to some user.*
getProfiles(boolean filterByActive)	*List<Profile>*	*Return the list of all profiles.*
getUserProfile(String userId)	*Profile*	*Return the profile assigned to a user.*
setUserProfile(String userId, long profileId)	*void*	*Change the assigned profile to a user.*
isManageUsers()	*boolean*	*Returns a boolean that indicates if the user is on manager or not.*
isManageRoles()	*boolean*	*Returns a boolean that indicates if the role is on manager or not.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example of the Principal adapter implementation

The example retrieves users, roles and mails based on properties files.

```
package com.openkm.plugin.principal;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.List;
import java.util.Properties;
import java.util.StringTokenizer;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.core.Config;
import com.openkm.db.bean.Profile;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalAdapterException;

public class UsersRolesPrincipalAdapter extends BasePlugin implements PrincipalAdapte

    private static Logger log = LoggerFactory.getLogger(UsersRolesPrincipalAdapter.cl

    @Autowired
    private OKMAuth okmAuth;

    @Override
    public List<CommonUser> getUsers() throws PrincipalAdapterException {
        log.debug("getUsers()");
        List<CommonUser> list = new ArrayList<>();
        Properties prop = new Properties();

        try {
            prop.load(new FileInputStream(Config.HOME_DIR + "/server/default/conf/pro
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Enumeration<Object> e = prop.keys(); e.hasMoreElements();) {
            String user = (String) e.nextElement();
            if (!Config.SYSTEM_USER.equals(user)) {
                CommonUser commonUser = okmAuth.getUser(null, user);
                list.add(commonUser);
            }
        }

        log.debug("getUsers: {}", list);
        return list;
    }

    @Override
    public List<String> getRoles() throws PrincipalAdapterException {
        log.debug("getRoles()");
        List<String> list = new ArrayList<String>();
        Properties prop = new Properties();
```

```
        try {
            prop.load(new FileInputStream(Config.HOME_DIR + "/server/default/conf/pro
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Enumeration<Object> e = prop.elements(); e.hasMoreElements();) {
            for (StringTokenizer st = new StringTokenizer((String) e.nextElement(), "
                String role = st.nextToken();

                if (!Config.DEFAULT_ADMIN_ROLE.equals(role) && !list.contains(role))
                    list.add(role);
                }
            }
        }

        log.debug("getRoles: {}", list);
        return list;
    }

    @Override
    public CommonUser getUser(String userId) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getUser");
    }

    @Override
    public List<CommonUser> getUsersByRole(String role) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("getUsersByRole");
    }

    @Override
    public List<String> getRolesByUser(String user) throws PrincipalAdapterException
        throw new UnsupportedOperationException("getRolesByUser");
    }

    @Override
    public String getPassword(String user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getPassword");
    }

    @Override
    public CommonUser createUser(CommonUser user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("createUser");
    }

    @Override
    public void deleteUser(String user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("deleteUser");
    }

    @Override
    public CommonUser updateUser(CommonUser user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("updateUser");
    }

    @Override
    public void createRole(String role, boolean active) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("createRole");
    }

    @Override
    public void deleteRole(String role) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("deleteRole");
    }
```

```
    @Override
    public void updateRole(String role, boolean active) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("updateRole");
    }

    @Override
    public void assignRole(String user, String role) throws PrincipalAdapterException
        throw new UnsupportedOperationException("assignRole");
    }

    @Override
    public void removeRole(String user, String role) throws PrincipalAdapterException
        throw new UnsupportedOperationException("removeRole");
    }

    @Override
    public List<Profile> getProfiles(boolean filterByActive) throws PrincipalAdapterE
        throw new UnsupportedOperationException("getProfiles");
    }

    @Override
    public Profile getUserProfile(String userId) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getUserProfile");
    }

    @Override
    public void setUserProfile(String userId, long profileId) throws PrincipalAdapter
        throw new UnsupportedOperationException("setUserProfile");
    }

    @Override
    public boolean isManageUsers() {
        throw new UnsupportedOperationException("isManageUsers");
    }

    @Override
    public boolean isManageRoles() {
        throw new UnsupportedOperationException("isManageRoles");
    }

}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Rest Plugin (extending REST API)

The Rest Plugin feature helps you to extending the REST API.

You can create your own Form Validator

Conditions:

- *The new Rest class **must implement the "RestPlugin"** interface.*
- *The new Rest class **must be declared under the package "com.openkm.plugin.rest".***
- *The new Rest class **must be annotated with "@PluginImplementation"**.*
- *The new Rest class **must extend of "BasePlugin".***

Rest interface:

```
package com.openkm.plugin.rest;

import net.xeoh.plugins.base.Plugin;

import java.io.InputStream;
import java.util.Map;

/**
 * RestPlugin
 */
public interface RestPlugin extends Plugin {

    Object executePlugin(Map<String, String> parameters, InputStream is) throws Excep

}
```

*The **new class must be located into the package com.openkm.plugin.rest** because the application plugins system will try to load it from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system won't be able to use the class.*

More information at Register a new plugin.

Method description

Method	Type	Description
executePlugin	**String**	*The method executes a REST plugin an return an object.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example of REST plugin implementation

```
package com.openkm.plugin.rest;

import com.openkm.plugin.BasePlugin;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.io.InputStream;
import java.util.Map;

/**
 * Sample rest plugin
 */
@PluginImplementation
public class TestRestPlugin extends BasePlugin implements RestPlugin {

    private static Logger log = LoggerFactory.getLogger(TestRestPlugin.class);

    @Override
    public Object executePlugin(Map<String, String> parameters, InputStream is) throw
        log.debug("executePlugin({})", parameters);
        StringBuilder sb = new StringBuilder();
        sb.append("{");
        sb.append("className=").append(this.getClass().getCanonicalName());
        for (String key : parameters.keySet()) {
            sb.append(", " + key + "=").append(parameters.get(key));
        }
        return sb.toString();
    }
}
```

Example of REST plugin implementation what returns file

The sample below goes in combination of REST webservice "/rest/plugin/executeGetPluginReturnFile" method.

```
package com.openkm.plugin.rest;

import com.openkm.bean.Document;
import com.openkm.module.db.DbDocumentModule;
import com.openkm.plugin.BasePlugin;
import com.openkm.util.PathUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.core.io.InputStreamResource;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;

import java.io.InputStream;
import java.util.Map;

/**
 * Sample rest plugin
 */
@PluginImplementation
public class TestGetDocumentRestPlugin extends BasePlugin implements RestPlugin {
```

```
    private static Logger log = LoggerFactory.getLogger(TestGetDocumentRestPlugin.cla

    @Autowired
    private DbDocumentModule dbDocumentModule;

    @Autowired
    private PathUtils pathUtils;

    @Override
    public Object executePlugin(Map<String, String> parameters, InputStream is) throw
        log.debug("executePlugin({})", parameters);
        String docId = parameters.get("docId");
        boolean inline = parameters.containsKey("inline");
        Document doc = dbDocumentModule.getProperties(null, docId);
        String mimeType = doc.getMimeType();
        String fileName = pathUtils.getName(doc.getPath());
        InputStream isContent = dbDocumentModule.getContent(null, docId, false);

        HttpHeaders responseHeaders = new HttpHeaders();
        responseHeaders.add("Content-Type", mimeType);

        // inline true when you want to embedded the content into
        if (inline) {
            responseHeaders.add("Content-disposition", "inline; filename=\"" + fileNa
        } else {
            responseHeaders.add("Content-Disposition", "attachment; filename=\"" + fi
        }

        responseHeaders.setContentLength(doc.getActualVersion().getSize());
        InputStreamResource inputStreamResource = new InputStreamResource(isContent);
        log.debug("TestGetDocumentRestPlugin: [BINARY]");
        return new ResponseEntity<>(inputStreamResource, responseHeaders, HttpStatus.
    }
}
```

Objects declaration

The method executionPlugin returns an Object. This object will be processed - marshall - by OpenKM. Basic types like String, Integer, etc. can be directly used, but for complex objects you need to use annotation on classes.

Marshall and unmarshall strategies:

- *The result of the executed method is always an String. Internally the method converts the complex objects to JSON.*
- *The result of the executed mehod is a complex Object.*

String scenario

From the plugin side - server side - you must convert the Object to JSON:

```
return new Gson().toJson(someObject)
```

From the cliend side you must convert JSON String to Object:

```
// value contains the returned String value by the REST call
SomeObject someObject = new Gson().fromJson(value, SomeObject.class);
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Complex Object scenario

You can use existing OpenKM classes like Document, Folder, Record, Mail among others, because theses clases have already the annotation.

Use the anotation (in the sample below the object will be marshalled with name "mail"):

@XmlRootElement(name = "mail")

Example of complex classes:

Return a list of Activities:

```
package com.openkm.ws.rest.util;

import com.openkm.db.bean.Activity;

import java.util.ArrayList;
import java.util.List;

public class ActivityList {

    List<Activity> activities = new ArrayList<>();

    private int totalElements;

    public List<Activity> getActivities() {
        return activities;
    }

    public void setActivities(List<Activity> activities) {
        this.activities = activities;
    }

    public int getTotalElements() {
        return totalElements;
    }

    public void setTotalElements(int totalElements) {
        this.totalElements = totalElements;
    }
}
```

Complex object:

```
package com.openkm.bean;

public class Record extends Parent {

    private static final long serialVersionUID = 1L;
    public static final String TYPE = "okm:record";
    public static final String NAME = "okm:name";

    private boolean locked;
    private LockInfo lockInfo;
    private String title;
    private boolean hasLinks;
```

```
    public boolean isLocked() {
        return locked;
    }

    public void setLocked(boolean locked) {
        this.locked = locked;
    }

    public LockInfo getLockInfo() {
        return lockInfo;
    }

    public void setLockInfo(LockInfo lockInfo) {
        this.lockInfo = lockInfo;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public boolean hasLinks() {
        return hasLinks;
    }

    public void setHasLinks(boolean hasLinks) {
        this.hasLinks = hasLinks;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("{");
        sb.append("uuid=").append(uuid);
        sb.append(", parent=").append(parent);
        sb.append(", path=").append(path);
        sb.append(", permissions=").append(permissions);
        sb.append(", created=").append(created == null ? null : created.getTime());
        sb.append(", subscribed=").append(subscribed);
        sb.append(", subscriptors=").append(subscriptors);
        sb.append(", keywords=").append(keywords);
        sb.append(", categories=").append(categories);
        sb.append(", notes=").append(notes);
        sb.append(", locked=").append(locked);
        sb.append(", lockInfo=").append(lockInfo);
        sb.append(", title=").append(title);
        sb.append(", nodeClass=").append(getNodeClass());
        sb.append(", nodeClassChildren=").append(getNodeClassChildren());
        sb.append(", dispositionCurrentStage=").append(getDispositionCurrentStage());
        sb.append("}");
        return sb.toString();
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Suggestion Analyzer

You can create your own Suggestion Analyzer.

Conditions:

- *The new Suggestion Analyzer class **must implement the "Suggestion"** interface.*
- *The new Suggestion Analyzer class **must be declared under the package "com.openkm.plugin.form.suggestion".***
- *The new Suggestion Analyzer class **must be annotated with "@PluginImplementation".***
- *The new Suggestion Analyzer class **must extend of "BasePlugin".***

Suggestion Analyzer interface:

```
package com.openkm.plugin.form.suggestion;

import com.openkm.bean.form.Select;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface Suggestion extends Plugin {
    List<String> getSuggestions(String nodeUuid, String nodePath, Select sel) throws
        DatabaseException;
}
```

*The **new class must be loaded into the package com.openkm.plugin.form.suggestion** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

Methods description

Method	*Type*	*Description*
getSuggestions(String nodeUuid, String nodePath, Select sel)	*List<String>*	*Return a list of suggested words to tag the document.* *The method need the document uuid (nodeUuid) and the document path (nodePath).* *Select can be used as a collection of possible*

		values (enclosed dictionary).

Example of Suggestion Analyzer

```
package com.openkm.plugin.form.suggestion;

import com.openkm.bean.form.Option;
import com.openkm.bean.form.Select;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.db.bean.NodeDocumentVersion;
import com.openkm.db.service.NodeDocumentSrv;
import com.openkm.db.service.NodeDocumentUtilsSrv;
import com.openkm.db.service.NodeDocumentVersionSrv;
import com.openkm.extractor.TextExtractorWork;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalUtils;
import com.openkm.search.lucene.LuceneUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

@PluginImplementation
public class DocumentTermsSuggestion extends BasePlugin implements Suggestion {
    private static final Logger log = LoggerFactory.getLogger(DocumentTermsSuggestion

    @Autowired
    private NodeDocumentVersionSrv nodeDocumentVersionSrv;

    @Autowired
    private NodeDocumentUtilsSrv nodeDocumentUtilsSrv;

    @Autowired
    private NodeDocumentSrv nodeDocumentSrv;

    @Autowired
    private LuceneUtils luceneUtils;

    @Override
    public List<String> getSuggestions(String nodeUuid, String nodePath, Select sel)
   DatabaseException {
        log.debug("getSuggestions({}, {}, {})", nodeUuid, nodePath, sel);
        List<String> list = new ArrayList<>();

        try {
            if (nodeDocumentSrv.itemExists(nodeUuid)) {
                // Get text extraction
                if (!nodeDocumentUtilsSrv.isTextExtracted(nodeUuid)) {
                    // Force text extraction
                    NodeDocumentVersion nDocVer = nodeDocumentVersionSrv.findCurrentV
                    TextExtractorWork tew = new TextExtractorWork();
                    tew.setDocUuid(nodeUuid);
                    tew.setDocPath(nodePath);
                    tew.setDocVerUuid(nDocVer.getUuid());
                    tew.setTenant(PrincipalUtils.getTenantId());

                    // Execute extractor
                    nodeDocumentUtilsSrv.textExtractorHelper(tew);
```

```
                }

                for (String term : luceneUtils.getTerms(nodeUuid)) {
                    for (Option opt : sel.getOptions()) {
                        if (term.equals(opt.getLabel().toLowerCase())) {
                            if (!list.contains(opt.getValue())) {
                                list.add(opt.getValue());
                                break;
                            }
                        }
                    }
                }
            }
        } catch (IOException e) {
            throw new SuggestionException(e.getMessage(), e);
        }

        return list;
    }
}
```

Creating your own Suggestbox plugin

You can create your own Suggestbox

Conditions:

- *The new Suggestbox class **must implement the "SuggestBoxValues"** interface.*
- *The new Suggestbox class **must be declared under the package "com.openkm.plugin.form.values".***
- *The new Suggestbox class **must be annotated with "@PluginImplementation"**.*
- *The new Suggestbox class **must extend of "BasePlugin".***

Suggestbox interface:

```
package com.openkm.plugin.form.values;

import com.openkm.db.bean.KeyValue;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface SuggestBoxValues extends Plugin {

    String getName();

    List<KeyValue> getSuggestBoxValuesByFilter(String filterValue) throws SuggestBoxE

    KeyValue getExactKeyValueByClassName(String value) throws SuggestBoxException;

}
```

*The **new class must be loaded into the package com.openkm.plugin.form.values** because application plugin system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at: Register a new plugin.

Methods description

Method	*Type*	*Description*
getName()	*String*	*Return the name what will be shown at plugins table.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getSuggestBoxValuesByFilter(String filterValue)	*List<KeyValue>*	*Return a list with the object KeyValue.*
getExactKeyValueByClassName(String value)	*KeyValue*	*Return a object KeyValue.*

Example of implementation

```
package com.openkm.plugin.form.values;

import net.xeoh.plugins.base.annotations.PluginImplementation;

import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.db.bean.KeyValue;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalAdapterException;

@PluginImplementation
public class SuggestBoxUserList extends BasePlugin implements SuggestBoxValues {

    @Autowired
    private OKMAuth okmAuth;

    @Override
    public String getName() {
        return "User list suggest box";
    }

    @Override
    public List<KeyValue> getSuggestBoxValuesByFilter(String filterValue) throws Sugg
        List<KeyValue> list = new ArrayList<>();

        try {
            for (CommonUser user : okmAuth.getUsers(null)) {
                if (user.getName().toLowerCase().contains(filterValue.toLowerCase()))
                    KeyValue value = new KeyValue();
                    value.setKey(user.getId());
                    value.setValue(user.getName());
                    list.add(value);
                }
            }
        } catch (PrincipalAdapterException e) {
            throw new SuggestBoxException(e);
        }

        return list;
    }

    @Override
    public KeyValue getExactKeyValueByClassName(String value) throws SuggestBoxExcept
        KeyValue keyValue = null;

        try {
            for (CommonUser user : okmAuth.getUsers(null)) {
                if (user.equals(value)) {
```

```
                    keyValue = new KeyValue();
                    keyValue.setKey(user.getId());
                    keyValue.setValue(user.getName());
                    break;
                }
            }
        } catch (PrincipalAdapterException e) {
            throw new SuggestBoxException(e);
        }

        return keyValue;
    }
}
```

Creating your own Stack Metadata Adapter

The StackMetadata class implements the logic that takes control on what is shown into metadata stack - tree nodes on left panel - and their objects (documents, nodes, folders, records) on right panel (browser).

You can create your own Stack Metadata Adapter.

Conditions:

- *The new Stack Metadata Adapter class* ***must implement the "StackMetadataAdapter"*** *interface.*
- *The new Stack Metadata Adapter class* ***must be declared under the package "com.openkm.plugin.stack.metadata".***
- *The new Stack Metadata Adapter class* ***must be annotated with "@PluginImplementation".***
- *The new Stack Metadata Adapter class* ***must extend of "BasePlugin".***

Stack Metadata Adapter interface:

```
package com.openkm.plugin.stack;

import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.bean.Record;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface StackMetadataAdapter extends Plugin {

    String getName();

    boolean isPropertyGroupHidden(String pgName);

    Folder getProperties(String fldPath);

    List<Folder> getChildren(String fldPath);

    List<Document> getDocumentMetadataChildren(String fldPath);

    List<Record> getRecordMetadataChildren(String fldPath);

    List<Mail> getMailMetadataChildren(String fldPath);

    List<Folder> getFolderMetadataChildren(String fldPath);
}
```

The ***new class must be loaded into the package com.openkm.plugin.stack.metadata*** *because application plugins system will try to load from there.*

Do not miss the tag ***@PluginImplementation*** *otherwise the application plugin system will not be able to retrieve the new class.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

More information at Register a new plugin.

Methods description

Method	*Type*	*Description*
getName()	*String*	*Return the name shown at plugins table.*
isPropertyGroupHidden(String pgName)	*boolean*	*Return if metadata fields are hidden.*
getProperties(String fldPath)	*Folder*	*Return folder properties.*
getChildren(String fldPath)	*List<Folder>*	*Return a list of folders.* *When folder node is chosen from tree (left panel) this method is called to retrieve children folders. These folders are displayed on tree as child nodes.*
getDocumentMetadataChildren(String fldPath)	*List<Document>*	*Return a list of documents.* *When the folder node is chosen from tree (left panel) this method is called to retrieve children documents. These documents are displayed on browser explorer, right panel.*
getRecordMetadataChildren(String fldPath)	*List<Record>*	*Return a list of records.* *When folder node is chosen from tree (left panel) this method is called to retrieve children records. These records are displayed on browser explorer, right panel.*
getMailMetadataChildren(String fldPath)	*List<Mail>*	*Return a list of mails.* *When folder node is chosen from tree (left panel) this method is called to retrieve children mails. These mails are displayed on browser explorer, right panel.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getFolderMetadataChildren(String fldPath)	*List<Folder>*	*Return a list of folders.* *When folder node is chosen from tree (left panel) this method is called to retrieve children folders. These folders are displayed on browser explorer, right panel.*

Example of the Stack Metadata implementation

```
package com.openkm.plugin.stack.metadata;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.api.OKMRepository;
import com.openkm.api.OKMSearch;
import com.openkm.bean.*;
import com.openkm.bean.form.*;
import com.openkm.core.Config;
import com.openkm.db.bean.KeyValue;
import com.openkm.db.service.KeyValueSrv;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.stack.StackMetadataAdapter;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.List;

@PluginImplementation
public class ParentChildStackMetadataAdapter extends BasePlugin implements StackMetad

    private static Logger log = LoggerFactory.getLogger(ParentChildStackMetadataAdapt

    @Autowired
    private OKMPropertyGroup okmPropertyGroup;

    @Autowired
    private OKMRepository okmRepository;

    @Autowired
    private KeyValueSrv keyValueSrv;

    @Autowired
    private OKMSearch okmSearch;

    @Override
    public String getName() {
        return "Parent Child Metadata Adapter";
    }

    @Override
    public List<Folder> getFolderMetadataChildren(String fldPath) {
        List<Folder> folderList = new ArrayList<>();

        try {
            // Metadata value level
            if (fldPath.startsWith("/" + Repository.METADATA) && (fldPath.split("/").
                String subFolder[] = fldPath.split("/");
```

```
                String group = subFolder[2];
                String property = subFolder[subFolder.length - 2];
                String value = subFolder[subFolder.length - 1];

                // In this case we are always in a select with parent Element
                // We need to find de property name, because in property we have the
                if (subFolder.length - 1 > 4) {
                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : ((Select) formElement).getOption
                                    if (option.getValue().equalsIgnoreCase(value)) {
                                        property = select.getName();
                                        break;
                                    }
                                }
                            }
                        }
                    }
                }

                folderList = okmSearch.getFoldersByPropertyValue(null, group, propert
            }
        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }

        return folderList;
    }

    @Override
    public List<Document> getDocumentMetadataChildren(String fldPath) {
        List<Document> docList = new ArrayList<>();

        try {
            // Case metadata at value level
            if (fldPath.split("/").length - 1 >= 4) {
                String subFolder[] = fldPath.split("/");
                String group = subFolder[2];
                String property = subFolder[subFolder.length - 2];
                String value = subFolder[subFolder.length - 1];

                // In this case we are always in a select with parent Element
                // We need to find de property name, because we have the parent value
                if (subFolder.length - 1 > 4) {
                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : ((Select) formElement).getOption
                                    if (option.getValue().equalsIgnoreCase(value)) {
                                        property = select.getName();
                                        break;
                                    }
                                }
                            }
                        }
                    }
                }
                docList = okmSearch.getDocumentsByPropertyValue(null, group, property
            }

        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }
```

```
        return docList;
    }

    @Override
    public List<Record> getRecordMetadataChildren(String fldPath) {
        List<Record> recordList = new ArrayList<>();

        try {
            // Case metadata at value level
            if (fldPath.split("/").length - 1 >= 4) {
                String subFolder[] = fldPath.split("/");
                String group = subFolder[2];
                String property = subFolder[subFolder.length - 2];
                String value = subFolder[subFolder.length - 1];

                // In this case we are always in a select with parent Element
                // We need to find de property name, because we have the parent value
                if (subFolder.length - 1 > 4) {
                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : ((Select) formElement).getOption
                                    if (option.getValue().equalsIgnoreCase(value)) {
                                        property = select.getName();
                                        break;
                                    }
                                }
                            }
                        }
                    }
                }

                recordList = okmSearch.getRecordsByPropertyValue(null, group, propert
            }

        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }

        return recordList;
    }

    @Override
    public List<Mail> getMailMetadataChildren(String fldPath) {
        List<Mail> mailList = new ArrayList<>();

        try {
            // Case metadata at value level
            if (fldPath.split("/").length - 1 >= 4) {
                String subFolder[] = fldPath.split("/");
                String group = subFolder[2];
                String property = subFolder[subFolder.length - 2];
                String value = subFolder[subFolder.length - 1];

                // In this case we are always in a select with parent Element
                // We need to find de property name, because we have the parent value
                if (subFolder.length - 1 > 4) {
                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : ((Select) formElement).getOption
                                    if (option.getValue().equalsIgnoreCase(value)) {
                                        property = select.getName();
                                        break;
                                    }
```

```
                                }
                            }
                        }
                    }
                }

                mailList = okmSearch.getMailsByPropertyValue(null, group, property, v
            }

        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }

        return mailList;
    }

    @Override
    public List<Folder> getChildren(String fldPath) {
        List<Folder> folderList = new ArrayList<>();

        try {
            switch (fldPath.split("/").length - 1) {
                case 1: // getting group level
                    for (PropertyGroup pg : okmPropertyGroup.getAllGroups(null)) {
                        // Only represent metadata groups which have some representab
                        boolean hasValidChildren = false;

                        for (FormElement formElement : okmPropertyGroup.getPropertyGr
                            if (!isPropertyGroupHidden(pg.getName())) {
                                if (!isPropertyGroupHidden(formElement.getName()) &&
                                    hasValidChildren = true;
                                    break;
                                }
                            }
                        }

                        if (hasValidChildren) {
                            Folder folder = new Folder();
                            String path = fldPath + "/" + pg.getName();
                            folder.initMetadata(path, pg.getLabel(),
                                    okmPropertyGroup.getPropertyGroupForm(null, pg.ge
                            folderList.add(folder);
                        }
                    }
                    break;

                case 2: // getting property level
                    String grpName = fldPath.substring(fldPath.lastIndexOf("/") + 1);
                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (!isPropertyGroupHidden(formElement.getName())) {
                            if (formElement instanceof Select) {
                                Select select = (Select) formElement;
                                // Only draw select with no parent
                                if (select.getParentElement() == null || select.getPa
                                    Folder folder = new Folder();
                                    String path = fldPath + "/" + formElement.getName
                                    folder.initMetadata(path, formElement.getLabel(),
                                    folderList.add(folder);
                                }
                            } else if (formElement instanceof SuggestBox || formEleme
                                Folder folder = new Folder();
                                String path = fldPath + "/" + formElement.getName();
                                folder.initMetadata(path, formElement.getLabel(), fal
                                folderList.add(folder);
                            }
                        }
                    }
```

```
                    break;

                case 3: // getting property value first level
                    String subFolder[] = fldPath.split("/");
                    grpName = subFolder[2];
                    String propertyName = subFolder[3];

                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement.getName().equals(propertyName)) {
                            if (formElement instanceof Select) {
                                for (Option option : ((Select) formElement).getOption
                                    Folder folder = new Folder();
                                    String path = fldPath + "/" + option.getValue();
                                    folder.initMetadata(path, option.getLabel(), fals
                                    folderList.add(folder);
                                }
                            } else if (formElement instanceof SuggestBox) {
                                SuggestBox sb = ((SuggestBox) formElement);
                                String sqlFilter = sb.getFilterQuery();

                                // replacing filter parameter, if exist
                                sqlFilter = sqlFilter.replaceAll("\\{0\\}", "");

                                for (KeyValue keyValue : keyValueSrv.getKeyValues(sb.
                                    Folder folder = new Folder();
                                    String path = fldPath + "/" + keyValue.getKey();
                                    folder.initMetadata(path, keyValue.getValue(), fa
                                    folderList.add(folder);
                                }
                            } else if (formElement instanceof CheckBox) {
                                Folder folder = new Folder();
                                String path = fldPath + "/true";
                                folder.initMetadata(path, "true", false);
                                folderList.add(folder);
                                folder = new Folder();
                                path = fldPath + "/false";
                                folder.initMetadata(path, "false", false);
                                folderList.add(folder);
                            }
                            break;
                        }
                    }
                    break;

                default: // getting property value for select with parent elements
                    subFolder = fldPath.split("/");
                    grpName = subFolder[2];
                    String pgValue = subFolder[subFolder.length - 1];

                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : select.getOptions()) {
                                    if (option.getParentValue().equalsIgnoreCase(pgVa
                                        Folder folder = new Folder();
                                        String path = fldPath + "/" + option.getValue
                                        folder.initMetadata(path, option.getLabel(),
                                        folderList.add(folder);
                                    }
                                }
                            }
                        }
                    }
                    break;
            }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }

        return folderList;
    }

    /**
     * If if a given property or property group is hidden.
     *
     * @param pgName The name of the property or property group.
     * @return Is property group is hidden or not by Config.
     */
    @Override
    public boolean isPropertyGroupHidden(String pgName) {
        boolean isHidden = false;

        for (String metadata : Config.STACK_METADATA_HIDDEN_PROPERTIES) {
            if (pgName.equalsIgnoreCase(metadata)) {
                isHidden = true;
                break;
            }
        }

        return isHidden;
    }

    @Override
    public Folder getProperties(String fldPath) {
        Folder folder = new Folder();

        try {
            String grpName = "";
            String propertyName = "";
            String subFolder[] = fldPath.split("/");

            switch (fldPath.split("/").length - 1) {
                case 1:
                    folder = okmRepository.getMetadataFolder(null);
                    break;

                case 2: // group level
                    grpName = fldPath.substring(fldPath.lastIndexOf("/") + 1);
                    String label = "";

                    for (PropertyGroup pg : okmPropertyGroup.getAllGroups(null)) {
                        if (pg.getName().equals(grpName)) {
                            label = pg.getLabel();
                            break;
                        }
                    }

                    List<FormElement> props = okmPropertyGroup.getPropertyGroupForm(n
                    folder.initMetadata(fldPath, label, props.size() > 0);
                    break;

                case 3: // property level
                    grpName = subFolder[2];
                    propertyName = subFolder[3];

                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement.getName().equals(propertyName)) {
                            folder.initMetadata(fldPath, formElement.getLabel(), fals
                            break;
                        }
                    }
                    break;
```

```
                case 4: // value level
                    grpName = subFolder[2];
                    propertyName = subFolder[3];
                    String value = subFolder[4];

                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement.getName().equals(propertyName)) {
                            if (formElement instanceof Select) {
                                Select select = (Select) formElement;
                                // Only select with no parent
                                if (select.getParentElement() == null || select.getPa
                                    for (Option option : ((Select) formElement).getOp
                                        if (option.getValue().equals(value)) {
                                            folder.initMetadata(fldPath, option.getLa
                                            break;
                                        }
                                    }
                                }
                            } else if (formElement instanceof SuggestBox) {
                                SuggestBox sb = ((SuggestBox) formElement);
                                String sqlFilter = sb.getFilterQuery();
                                sqlFilter = sqlFilter.replaceAll("\\{0\\}", ""); // r

                                for (KeyValue keyValue : keyValueSrv.getKeyValues(sb.
                                    if (keyValue.getKey().equals(value)) {
                                        folder.initMetadata(fldPath, keyValue.getValu
                                        break;
                                    }
                                }
                            } else if (formElement instanceof CheckBox) {
                                folder.initMetadata(fldPath, value, false);
                            }
                            break;
                        }
                    }
                    break;

                default: // value level for select with parent
                    grpName = subFolder[2];
                    String pgValue = subFolder[subFolder.length - 1];

                    for (FormElement formElement : okmPropertyGroup.getPropertyGroupF
                        if (formElement instanceof Select) {
                            Select select = (Select) formElement;
                            // Only select with no parents
                            if (select.getParentElement() != null && !select.getParen
                                for (Option option : ((Select) formElement).getOption
                                    if (option.getValue().equalsIgnoreCase(pgValue))
                                        folder.initMetadata(fldPath, option.getLabel(
                                        break;
                                    }
                                }
                            }
                        }
                    }
                    break;
            }
        } catch (Exception e) {
            log.error(e.getMessage(), e);
        }

        return folder;
    }

}
```

?

?

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own Text extractor

You can create your own Text Extractor

Conditions:

- *The new Text Extractor class **must implement the "FieldValidator"** interface.*
- *The new Text Extractor class **must be declared under the package "com.openkm.plugin.extractor".***
- *The new Text Extractor class **must be annotated with "@PluginImplementation"**.*
- *The new Text Extractor class **must extend of "AbstractTextExtractor".***

Text Extractor interface:

```
package com.openkm.plugin.extractor;

import net.xeoh.plugins.base.Plugin;

import java.io.IOException;
import java.io.InputStream;

public interface TextExtractor extends Plugin {

    String[] getContentTypes();

    String extractText(InputStream stream, String type, String encoding) throws IOExc
}
```

*The **new class must be loaded into the package com.openkm.plugin.extractor** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

Methods description

Method	*Type*	*Description*
getContentTypes()	*String[]*	*Returns the MIME types supported by this extractor. The returned strings must be in lower case, and the returned array must not be empty.*
extractText(InputStream stream, String type, String	*String*	*Returns a reader for the text content of the given binary document. The content type and character encoding (if available and applicable) are given*

encoding)		*as arguments.*

Example of the Text extractor implementation

```
package com.openkm.plugin.extractor;

import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.apache.commons.io.IOUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.nio.charset.StandardCharsets;

/**
 * Text extractor for plain text.
 */
@PluginImplementation
public class PlainTextExtractor extends AbstractTextExtractor {

    private static final Logger log = LoggerFactory.getLogger(PlainTextExtractor.clas

    /**
     * Creates a new <code>PlainTextExtractor</code> instance.
     */
    public PlainTextExtractor() {
        super(new String[]{"text/plain"});
    }

    // ------------------------------------------------------------< TextExtractor >
    /**
     * Wraps the given input stream to an {@link InputStreamReader} using the
     * given encoding, or the platform default encoding if the encoding is not
     * given or is unsupported. Closes the stream and returns an empty reader if
     * the given encoding is not supported.
     *
     * @param stream binary stream
     * @param type ignored
     * @param encoding character encoding, optional
     * @return reader for the plain text content
     * @throws IOException if the binary stream can not be closed in case of an
     * encoding issue
     */
    @Override
    public String extractText(InputStream stream, String type, String encoding) throw
        log.debug("extractText({}, {}, {})", stream, type, encoding);

        try {
            if (encoding != null) {
                return IOUtils.toString(stream, encoding);
            }
        } catch (UnsupportedEncodingException e) {
            log.warn("Unsupported encoding '{}', using default ({}) instead.", encodi
        }

        return IOUtils.toString(stream, StandardCharsets.UTF_8);
    }
}
```

Creating your own Version Number Adapter

The version number adapter is used by the application to set the logic that takes control of document version number each time the document is updated.

Keep in mind changing version number adapter when application already has document version it means sometimes you must migrate older version numbers to the new format or a new implementation must take in consideration older ones.

You can create your own Version Number Adapter

Conditions:

- *The new Form Validator class **must implement the "VersionNumerationAdapter"** interface.*
- *The new Form Validator class **must be declared under the package "com.openkm.plugin.vernum".***
- *The new Form Validator class **must be annotated with "@PluginImplementation"**.*
- *The new Form Validator class **must extend of "BasePlugin".***

Version Number Adapter interface:

```
package com.openkm.plugin.vernum;

import com.openkm.db.bean.NodeVersion;
import net.xeoh.plugins.base.Plugin;

public interface VersionNumerationAdapter extends Plugin {

    int INCREASE_DEFAULT = 0;
    int INCREASE_MINOR = 1;
    int INCREASE_MAJOR = 2;
    int INCREASE_MAJOR_MINOR = 3;

    String qs = "from NodeVersion nv where nv.parent=:parent and nv.name=:name";

    String getInitialVersionNumber();

    int getVersionNumberIncrement();

    String getNextVersionNumber(NodeVersion nVer, int increment);
}
```

*The **new class must be loaded into the package com.openkm.plugin.vernum** because application plugin system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*To **enable the new Version number adapter** go to **Administration > Configuration parameters** > find the parameter named **version.numeration.adapter** and modify the value with **com.openkm.vernum.OwnNumerationAdapter***

*To take effect this change need to **restating application**.*

Methods description

Method	*Type*	*Description*
getInitialVersionNumber()	*String*	*Return the first version of a new document.* *First version usually will be "1.0".*
getVersionNumberIncrement()	*int*	*Return a number that indicate the available increase options.* *Available values:* • *VersionNumerationAdapter.INCREASE_DEFAULT* • *VersionNumerationAdapter.INCREASE_MINOR* • *VersionNumerationAdapter.INCREASE_MAJOR* • *VersionNumerationAdapter.INCREASE_MAJOR_MINOR* *INCREASE_NONE: Not showing any option, increase by default.* *INCREASE_MINOR: Only shows the option of increase minor version, if not selected, increase by default.* *INCREASE_MAYOR: Only shows the option of increase mayor version, if not selected, increase by default.* *INCREASE_MAJOR_MINOR: Shows the options of increase major and minor version, if not selected, increase by default.*
String getNextVersionNumber(Session session, NodeDocument nDoc, NodeDocumentVersion nDocVer, int increment)()	*String*	*Calculate the next version number from a given one*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example of the Version number adapter implementation

```
package com.openkm.plugin.vernum;

import com.openkm.core.Config;
import com.openkm.db.bean.NodeVersion;
import com.openkm.db.repository.NodeVersionRepo;
import com.openkm.plugin.BasePlugin;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.springframework.beans.factory.annotation.Autowired;

@PluginImplementation
public class PlainVersionNumerationAdapter extends BasePlugin implements VersionNumer

    @Autowired
    private NodeVersionRepo nodeVersionRepo;

    @Override
    public String getInitialVersionNumber() {
        return String.format(Config.VERSION_NUMERATION_FORMAT, 1);
    }

    @Override
    public int getVersionNumberIncrement() {
        return INCREASE_DEFAULT;
    }

    @Override
    public String getNextVersionNumber(NodeVersion nVer, int increment) {
        String versionNumber = nVer.getName();
        int nextVerNumber = Integer.parseInt(versionNumber);
        NodeVersion nv;

        do {
            nextVerNumber++;
            String name = String.valueOf(nextVerNumber);
            nv = nodeVersionRepo.getNodeVersionByNodeAndName(nVer.getParent(), name);
        } while (nv != null);

        return String.format(Config.VERSION_NUMERATION_FORMAT, nextVerNumber);
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own security access manager

The security access manager is used by the OpenKM to check the user security access level of the nodes.

Conditions:

- *The new Principal Adapter class **must implement the "DbAccessManager"** interface.*
- *The new Principal Adapter class **must be declared under the package "com.openkm.plugin.access".***
- *The new Principal Adapter class **must be annotated with "@PluginImplementation".***
- *The new Principal Adapter class **must extend of "BasePlugin".***

Security access manager interface:

```
package com.openkm.plugin.access;

import com.openkm.core.AccessDeniedException;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.db.bean.NodeBase;
import com.openkm.principal.PrincipalAdapterException;
import net.xeoh.plugins.base.Plugin;

/**
 * Check user permissions on documents and folders.
 *
 * @author pavila
 */
public interface DbAccessManager extends Plugin {

 void checkPermission(NodeBase node, int permissions) throws AccessDeniedException, P

 boolean isGranted(NodeBase node, int permissions) throws DatabaseException, PathNotF

 boolean isGranted(NodeBase node, String user, int permissions) throws PrincipalAdapt

 boolean isGranted(NodeBase node, long nodeClass, int permissions) throws DatabaseExc
}
```

*The **new class must be loaded into the package com.openkm.plugin.access** because application plugins system will try to load from there.*

*Do not miss the tag **@PluginImplementation** otherwise, the application plugin system will not be able to retrieve the new class.*

More information at Register a new plugin.

*To **enable the new Security access manager** go to **Administration > Configuration parameters** > find the parameter named **security.access.manager** and modify the value with **com.openkm.plugin.access.OwnSecurityManager***

*To take effect this change we need to **restart the application**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Methods description

In the next table the values of permissions are (more information in the class Permission.java):

0 NONE
1 READ
2 WRITE
4 DELETE
8 SECURITY
16 MOVE
1024 DOWNLOAD
2048 START WORKFLOW
4096 COMPACT HISTORY
8192 METADATA

Method	Type	Description
checkPermission(NodeBase node, int permissions)	***void***	*Description*
isGranted(NodeBase node, int permissions)	***Boolean***	*Return true when the user in the session has the grant.*
isGranted(NodeBase node, String user, int permissions)	***Boolean***	*Return true when the user has the grant.*
isGranted(NodeBase node, long nodeClass, int permissions)	***Boolean***	*Return true when the user in the session has the grant.* *This method only has sense when file plan is enabled.*

Example of Security access manager implementation

```
package com.openkm.plugin.access;

import com.openkm.core.AccessDeniedException;
import com.openkm.core.Config;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.db.bean.NodeBase;
import com.openkm.db.bean.NodeClass;
import com.openkm.db.service.NodeBaseSrv;
import com.openkm.db.service.NodeClassSrv;
import com.openkm.module.common.CommonAuthModule;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.fileplan.security.FilePlanAccessManager;
import com.openkm.principal.PrincipalAdapterException;
import com.openkm.principal.PrincipalUtils;
import com.openkm.util.StackTraceUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
```

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;

/**
 * Check user permissions on documents and folders.
 *
 * @see com.openkm.plugin.access.DbRecursiveAccessManager
 */
@PluginImplementation
public class DbSimpleAccessManager extends BasePlugin implements DbAccessManager {
 private static final Logger log = LoggerFactory.getLogger(DbSimpleAccessManager.clas

 @Autowired
 private CommonAuthModule commonAuthModule;

 @Autowired
 private NodeClassSrv nodeClassSrv;

 @Autowired
 private NodeBaseSrv nodeBaseSrv;

 /**
  * Check for permissions.
  */
 public void checkPermission(NodeBase node, int permissions) throws AccessDeniedExcep
  if (!isGranted(node, permissions)) {
   String nodePath = nodeBaseSrv.getPathFromUuid(node.getUuid());
   throw new AccessDeniedException(node.getUuid() + " : " + nodePath);
  }
 }

 /**
  * Check for permissions.
  */
 @Override
 public boolean isGranted(NodeBase node, int permissions) throws DatabaseException {
  String userId = PrincipalUtils.getUser();
  Set<String> roles = PrincipalUtils.getRoles();
  return isGranted(node, userId, roles, permissions);
 }

 /**
  * Check for permissions.
  */
 @Override
 public boolean isGranted(NodeBase node, String user, int permissions) throws Princip
  List<String> roles = commonAuthModule.getRolesByUser(user);
  return isGranted(node, user, new HashSet<>(roles), permissions);
 }

 @Override
 public boolean isGranted(NodeBase node, long nodeClass, int permissions) throws Data
  NodeClass nc = nodeClassSrv.findByPk(nodeClass);
  FilePlanAccessManager accessManager = nodeClassSrv.findAccessManagerByClassName(nc.
  return accessManager.isGranted(node, nodeClass, PrincipalUtils.getUser(), Principal
 }

 /**
  * Check for permissions.
  */
 private boolean isGranted(NodeBase node, String user, Set<String> roles, int permiss
```

```
    log.debug("isGranted({}, {})", node.getUuid(), permissions);
    boolean access;

    if (user != null) {
      if (Config.SYSTEM_USER.equals(user) || Config.ADMIN_USER.equals(user)) {
        // An okmAdmin user has total access
        access = true;
      } else {
        if (roles.contains(Config.DEFAULT_ADMIN_ROLE)) {
          // An user with AdminRole has total access
          access = true;
        } else {
          access = checkProperties(node.getUserPermissions(), node.getRolePermissions(), u
        }
      }
    } else {
      access = true;

      log.info("*******************************************************");
      StackTraceUtils.logTrace(log);
      log.info("*******************************************************");
    }

    log.debug("isGranted: {}", access);
    return access;
  }

  /**
   * Check access properties
   */
  private boolean checkProperties(Map<String, Integer> usersPerms, Map<String, Integer
      Set<String> roles, int perms) {
    log.debug("checkProperties({}, {}, {}, {})", usersPerms, rolesPerms, roles, perms);
    boolean access = false;

    // Fist try with user permissions
    Integer userPerms = usersPerms.get(user);

    if (userPerms != null && (perms & userPerms) != 0) {
      log.debug("checkProperties: {}", true);
      return true;
    }

    // If there is no user specific access, try with roles permissions
    for (String role : roles) {
      Integer rolePerms = rolesPerms.get(role);

      if (rolePerms != null && (perms & rolePerms) != 0) {
        log.debug("checkProperties: {}", true);
        return true;
      }
    }

    log.debug("checkProperties: {}", access);
    return access;
  }
}
```

Creating templates

You can create new documents based on Templates. The application can handle 3 types of template documents:

- *PDF*
- *OpenOffice*
- *Text (This includes other text-based like HTML, for example).*

In order to generate the final document from the template, we need to make an association with Metadata . The final document will be assigned the corresponding Metadata with the user input values, so you can use them for a future search.

The template syntax, which is used by the FreeMarker Template Engine. This engine can be used from making simply variable substitution to loops and other control structure.

> *In the samples we use only the basic ones, but you can learn more at FreeMarker Manual.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating OpenOffice and LibreOffice templates

Advantages of using Office and LibreOffice templates:

- *Can produce odt file.*
- *Less time expended in the creation of the document.*

Disadvantages in comparison with Text templates:

- *Produce bigger documents.*
- *More difficulties in maintaining the templates in time.*

Preliminaries

Register the Metadata group definition at ***Administration*** *>* *Metadata*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Sample Template" name="okg:tpl">
    <input label="Name" name="okp:tpl.name"/>
    <input label="Birth Date" name="okp:tpl.birth_date" type="date" />
    <select label="Language" name="okp:tpl.language" type="simple">
      <option label="Java" value="java"/>
      <option label="Python" value="python"/>
      <option label="PHP" value="php" />
    </select>
  </property-group>
</property-groups>
```

In order to make a relation between the template form field and the property group, you have to name them following a naming convention: replace ":" and "." by "_". Here you can see the matching between both fields:

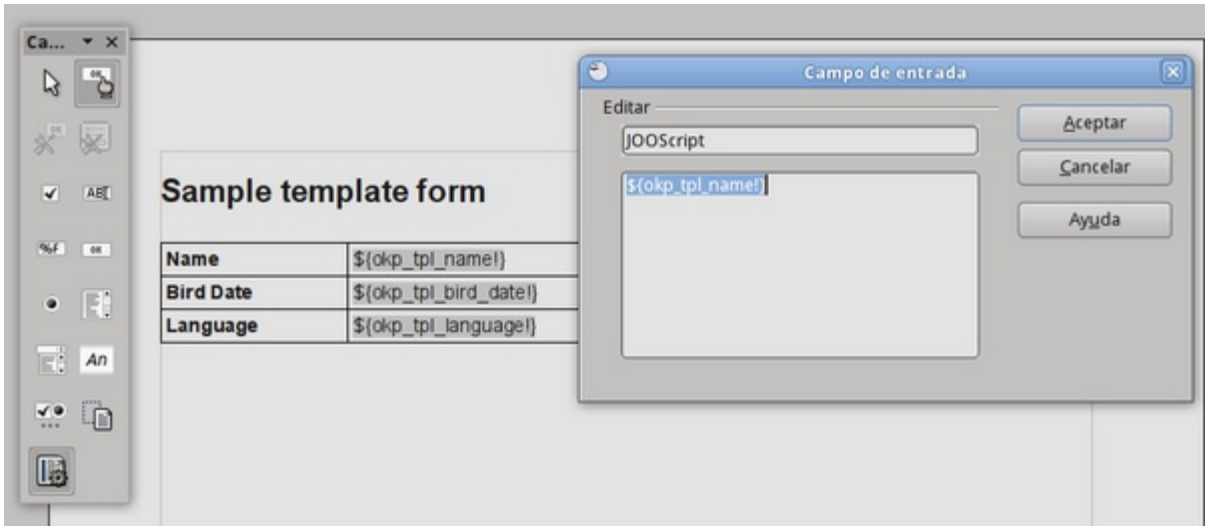
Metadata field name	*Template field name*	*Template field value*
okp:tpl.name	*okp_tpl_name*	*${okp_tpl_name!}*
okp:tpl.birth_date	*okp_tpl_birth_date*	*${okp_tpl_birth_date!}*
okp:tpl.language	*okp_tpl_language*	*${okp_tpl_language!}*

Creating the template

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*To create a template in OpenOffice.org go to **View** > **Toolbars** > **Form Controls**.*

The a new dialogue box will appear and you will be able to add form elements to the document. This dialogue box contains stuff such as Labels, Text Boxes, Check Boxes, etc.



The field that will be replaced with the value of a variable passed can be specified using the standard FreeMarker expression notation by typing the variable name into the document as follows:

```
Hi ${name}!
```

*However, mixing processing instructions and normal text into the document may become confusing and clutter the layout. So JODReports provides an alternative way of inserting fields. You can insert a visual field in OpenOffice.org Writer from the menu **Insert** / **Fields** / **Other...** (or just press **Ctrl+F2**), then click on the **Functions tab** and select **Input field**. Change the field Reference to **JOOScript**. In the following field Content enter the variable name (e.g. "$name"). This field will typically be displayed by Writer as a grayed rectangle with the variable name, while moving the mouse over it will reveal the field reference.*

This is the OpenOffice.org document used in this sample tpl.odt.

Read about OpenOffice.org templates at JODReports.

Date format

JODReports uses FreeMarker inside, so you can take advantage of the FreeMarker formatting flexibilities. For example you can set date format by this way:

```
${okp_tpl_birth_date?string("yyyy-MM-dd")}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

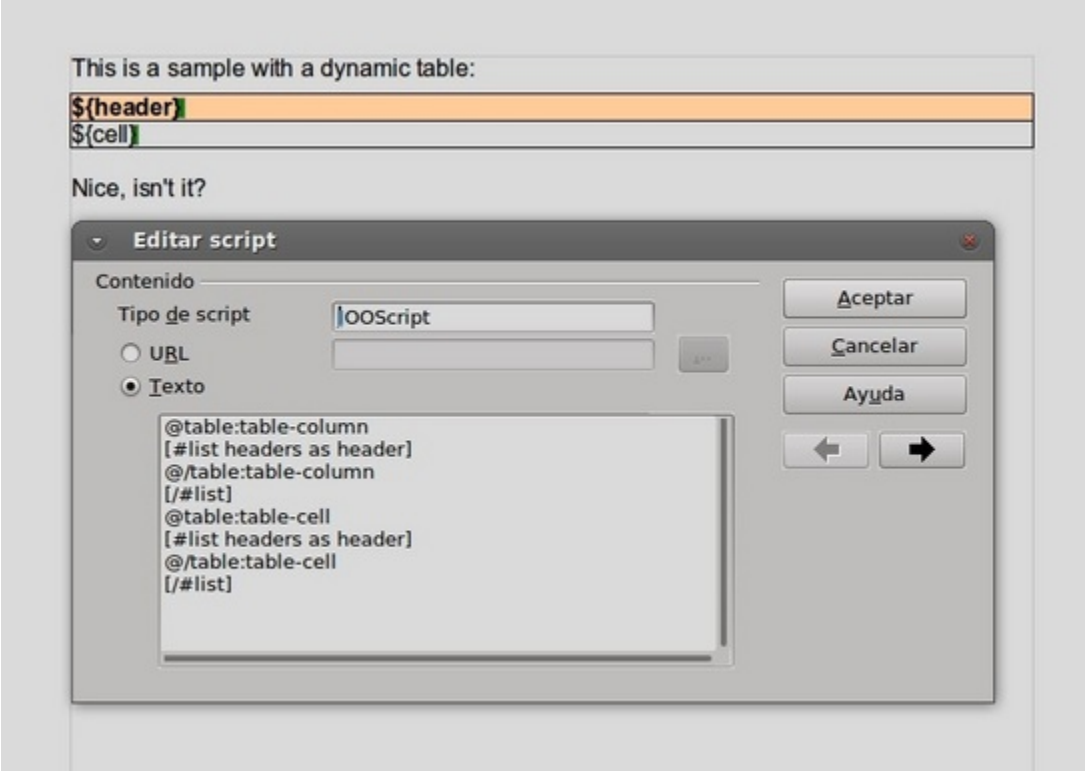
If date may be null, then you need to check this:

```
<#if okp_tpl_birth_date??>${okp_tpl_birth_date?string("yyyy-MM-dd")}</#if>
```

Learn more about FreeMarker formatting at [Built-in Reference](#).

Dynamic tables

You can even use the template feature to generate dynamic tables. Lets see an example.



In the screen shot you can see a document with a table. The header has a colored background and also is bold. The script shows the options to generate the table headers.

This is the OpenOffice.org document used in this sample: [dynamic_table.odt](#)

Learn more about this feature at:

- [JODReports: Inserting Directives at Special Locations](#).
- [Dynamic table column count](#)

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating PDF template

Advantages of using PDF templates:

- *Can produce PDF files.*

Disadvantages in comparison with Text templates:

- *Produce bigger documents.*
- *More difficulties in maintaining the templates in time.*
- *More time needed in the creation of the document (especially when the PDF optimization configuration parameter is enabled)*

You can create PDF template forms with several applications:

- *Acrobat Pro*
- *How-to: Create PDF forms with OpenOffice*
- *PDF Forms Designer*

Every PDF form field name needs to be unique, you can't set the same property value to several form fields.

When converting from OpenOffice document to PDF, repeated form fields are renamed to original_name_2, original_name_3, etc.

If you need to put the same property value in several fields, try Creating OpenOffice and LibreOffice templates.

In the sample below, we've used OpenOffice to create PDF template.

Preliminaries

Register the Metadata group definition at ***Administration >*** *Metadata.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Sample Template" name="okg:tpl">
    <input label="Name" name="okp:tpl.name"/>
    <input label="Birth Date" name="okp:tpl.birth_date" type="date" />
    <select label="Language" name="okp:tpl.language" type="simple">
      <option label="Java" value="java"/>
      <option label="Python" value="python"/>
      <option label="PHP" value="php" />
    </select>
  </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

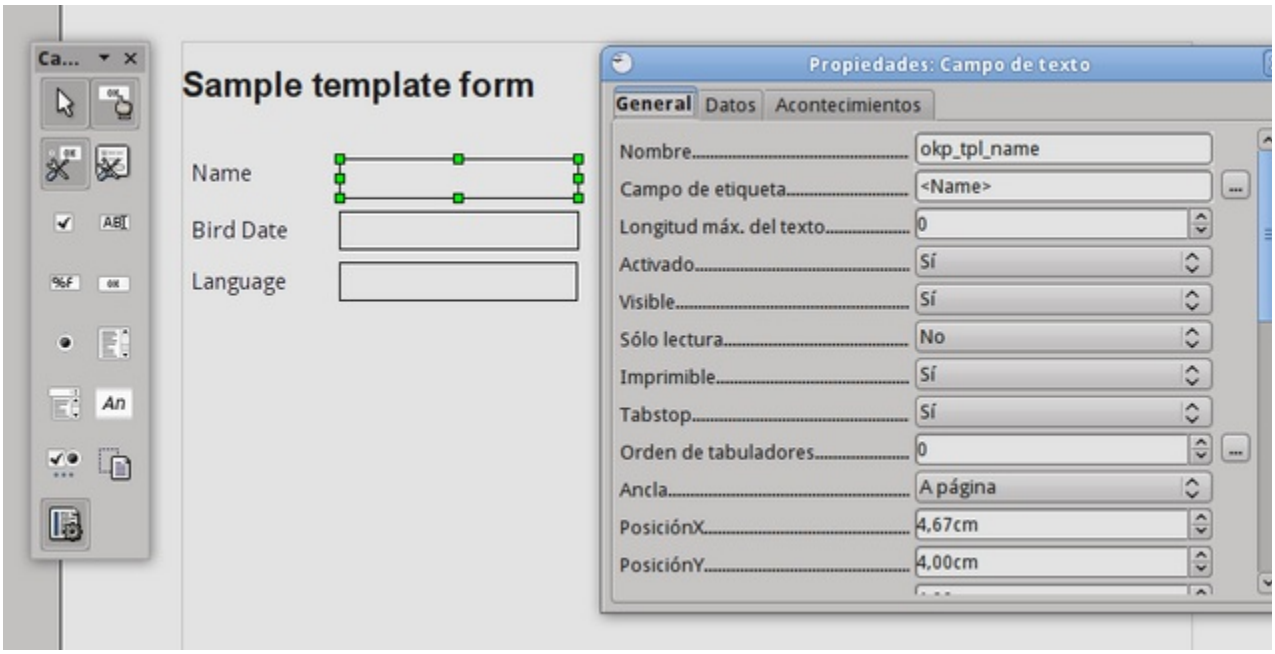
In order to make a relation between the template form field and the property group, you have to name them following a naming convention: replace ":" and "." by "_". Here you can see the matching between both fields:

Metadata field name	*Template field name*	*Template field value*
okp:tpl.name	*okp_tpl_name*	*${okp_tpl_name!}*
okp:tpl.birth_date	*okp_tpl_birth_date*	*${okp_tpl_birth_date!}*
okp:tpl.language	*okp_tpl_language*	*${okp_tpl_language!}*

Creating the template

To create a template in OpenOffice.org go to ***View*** *>* ***Toolbars*** *>* ***Form Controls***.

The a new dialogue will appear and you will be able to add form elements to the document. This dialogue contains stuff such as Labels, Text Boxes, Check Boxes, etc.



Once the form has been created, go to ***File*** *>* ***Export as PDF*** *to generate a PDF with these form fields. This is the final result:*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Here you have the generated PDF file tpl.pdf and this is the ODT source tpl_pdf.odt.

Date format

OpenKM uses FreeMarker inside, so you can take advantage of the FreeMarker formatting flexibilities. For example you can set date format by this way:

```
${okp_tpl_birth_date?string("yyyy-MM-dd")}
```

Learn more about FreeMarker formatting at Built-in Reference.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating Text templates

Advantages of using Text templates:

- *Can produce HTML or PDF files.*
- *The small size of the produced document.*
- *Easily maintain the templates in time.*
- *Less time expended in the creation of the document.*

Preliminaries

Register the Metadata group definition at ***Administration*** *>* *Metadata*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.7//EN"
                                 "http://www.openkm.com/dtd/property-groups-3.7.dtd">
<property-groups>
  <property-group label="Sample Template" name="okg:tpl">
    <input label="Name" name="okp:tpl.name"/>
    <input label="Birth Date" name="okp:tpl.birth_date" type="date" />
    <select label="Language" name="okp:tpl.language" type="simple">
      <option label="Java" value="java"/>
      <option label="Python" value="python"/>
      <option label="PHP" value="php" />
    </select>
  </property-group>
</property-groups>
```

In order to make a relation between the template form field and the property group, you have to name them following a naming convention: replace ":" and "." by "_". Here you can see the matching between both fields:

Metadata field name	*Template field name*	*Template field value*
okp:tpl.name	*okp_tpl_name*	*${okp_tpl_name!}*
okp:tpl.birth_date	*okp_tpl_birth_date*	*${okp_tpl_birth_date!}*
okp:tpl.language	*okp_tpl_language*	*${okp_tpl_language!}*

Creating the template

For this kind of template we will create an HTML one. This is the template source code:

```
<html>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
  <body>
    <h1>Sample template</h1>
    <table>
      <tr>
        <td><b>Name</b></td>
        <td>${okp_tpl_name!}</td>
      </tr>
      <tr>
        <td><b>Birth Date</b></td>
        <td>${okp_tpl_birth_date?string("yyyy-MM-dd")!}</td>
      </tr>
      <tr>
        <td><b>Language</b></td>
        <td>${okp_tpl_language!}</td>
      </tr>
    </table>
  </body>
</html>
```

*The tags **${okp_tpl_name!}**, **${okp_tpl_birth_date!}** and **${okp_tpl_language!}** will be replaced by the user input values. An error will occur and abort the template processing if you try to access a missing variable, but the ! operator handles this situation. In addition, you can put a default value in case of a mission one with this expression:*

```
${missing_value!"The default one"}
```

The result is an HTML file with the client input requested by OpenKM.

Date format

OpenKM uses FreeMarker inside, so you can take advantage of the FreeMarker formatting flexibilities. For example, you can set the date format in this way:

```
${okp_tpl_birth_date?string("yyyy-MM-dd")!}
```

Learn more about FreeMarker formatting at Built-in Reference.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own login module

The class must implement PrincipalAdapter interface.

```
package com.openkm.plugin.principal;

import com.openkm.bean.CommonUser;
import com.openkm.db.bean.Profile;
import com.openkm.principal.PrincipalAdapterException;
import net.xeoh.plugins.base.Plugin;

import java.util.List;

public interface PrincipalAdapter extends Plugin {

    /**
     * Method to retrieve all users from a authentication source.
     *
     * @return A Collection with all the users.
     * @throws PrincipalAdapterException If any error occurs.
     */
    List<CommonUser> getUsers() throws PrincipalAdapterException;

    /**
     * Method to retrieve all roles from a authentication source.
     *
     * @return A Collection with all the roles.
     * @throws PrincipalAdapterException If any error occurs.
     */
    List<String> getRoles() throws PrincipalAdapterException;

    /**
     * Method to retrieve user data
     *
     * @return the user data
     * @throws PrincipalAdapterException If any error occurs.
     */
    CommonUser getUser(String userId) throws PrincipalAdapterException;

    /**
     * Method to retrieve all users from a role.
     *
     * @return A Collection with all the users within a role.
     * @throws PrincipalAdapterException If any error occurs.
     */
    List<CommonUser> getUsersByRole(String role) throws PrincipalAdapterException;

    /**
     * Method to retrieve all roles from a user.
     *
     * @return A Collection with all the roles of the user.
     * @throws PrincipalAdapterException If any error occurs.
     */
    List<String> getRolesByUser(String user) throws PrincipalAdapterException;

    /**
     * Method to retrieve the user password
     *
     * @param user A user id.
     * @return The password of the user.
     * @throws PrincipalAdapterException If any error occurs.
     */
    String getPassword(String user) throws PrincipalAdapterException;
```

```
	/*
	* ---------------------------------------------------------------
	* These methods only works if using the OpenKM user database.
	* ---------------------------------------------------------------
	 */
	/**
	 * Method to create a new user
	 *
	 * @param user User data
	 * @return User
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	CommonUser createUser(CommonUser user) throws PrincipalAdapterException;

	/**
	 * Method to create a delete a user
	 *
	 * @param user A user id.
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	void deleteUser(String user) throws PrincipalAdapterException;

	/**
	 * Update user information
	 *
	 * @param user User data
	 * @return User
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	CommonUser updateUser(CommonUser user) throws PrincipalAdapterException;

	/**
	 * Method to create a new role
	 *
	 * @param role A role id.
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	void createRole(String role, boolean active) throws PrincipalAdapterException;

	/**
	 * Method to create a delete a role
	 *
	 * @param role A role id.
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	void deleteRole(String role) throws PrincipalAdapterException;

	/**
	 * Update role information
	 *
	 * @param role A role id..
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	void updateRole(String role, boolean active) throws PrincipalAdapterException;

	/**
	 * Method to assign a role
	 *
	 * @param user A user id.
	 * @param role A role id.
	 * @throws PrincipalAdapterException If any error occurs.
	 */
	void assignRole(String user, String role) throws PrincipalAdapterException;

	/**
	 * Method to remove a role
```

```
     *
     * @param user A user id.
     * @param role A role id.
     * @throws PrincipalAdapterException If any error occurs.
     */
    void removeRole(String user, String role) throws PrincipalAdapterException;

    /**
     * Get all profiles
     *
     * @param filterByActive
     * @return A list with all the registered profiles
     * @throws PrincipalAdapterException
     */
    List<Profile> getProfiles(boolean filterByActive) throws PrincipalAdapterExceptio

    /**
     * Get user profile
     *
     * @return The profile assigned to this user
     * @throws PrincipalAdapterException
     */
    Profile getUserProfile(String userId) throws PrincipalAdapterException;

    /**
     * Set the user profile
     *
     * @param userId The user id.
     * @param profileId The profile to be assigned.
     * @throws PrincipalAdapterException
     */
    void setUserProfile(String userId, long profileId) throws PrincipalAdapterExcepti

    boolean isManageUsers();

    boolean isManageRoles();
}
```

PrincipalAdapter sample based on properties

```
package com.openkm.plugin.principal;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.List;
import java.util.Properties;
import java.util.StringTokenizer;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.core.Config;
import com.openkm.db.bean.Profile;
import com.openkm.plugin.BasePlugin;
import com.openkm.principal.PrincipalAdapterException;

public class UsersRolesPrincipalAdapter extends BasePlugin implements PrincipalAdapte
```

```
    private static Logger log = LoggerFactory.getLogger(UsersRolesPrincipalAdapter.cl

    @Autowired
    private OKMAuth okmAuth;

    @Override
    public List<CommonUser> getUsers() throws PrincipalAdapterException {
        log.debug("getUsers()");
        List<CommonUser> list = new ArrayList<>();
        Properties prop = new Properties();

        try {
            prop.load(new FileInputStream(Config.HOME_DIR + "/server/default/conf/pro
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Enumeration<Object> e = prop.keys(); e.hasMoreElements();) {
            String user = (String) e.nextElement();
            if (!Config.SYSTEM_USER.equals(user)) {
                CommonUser commonUser = okmAuth.getUser(null, user);
                list.add(commonUser);
            }
        }

        log.debug("getUsers: {}", list);
        return list;
    }

    @Override
    public List<String> getRoles() throws PrincipalAdapterException {
        log.debug("getRoles()");
        List<String> list = new ArrayList<String>();
        Properties prop = new Properties();

        try {
            prop.load(new FileInputStream(Config.HOME_DIR + "/server/default/conf/pro
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Enumeration<Object> e = prop.elements(); e.hasMoreElements();) {
            for (StringTokenizer st = new StringTokenizer((String) e.nextElement(), "
                String role = st.nextToken();

                if (!Config.DEFAULT_ADMIN_ROLE.equals(role) && !list.contains(role))
                    list.add(role);
                }
            }
        }

        log.debug("getRoles: {}", list);
        return list;
    }

    @Override
    public CommonUser getUser(String userId) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getUser");
    }

    @Override
    public List<CommonUser> getUsersByRole(String role) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("getUsersByRole");
```

```
    }

    @Override
    public List<String> getRolesByUser(String user) throws PrincipalAdapterException
        throw new UnsupportedOperationException("getRolesByUser");
    }

    @Override
    public String getPassword(String user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getPassword");
    }

    @Override
    public CommonUser createUser(CommonUser user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("createUser");
    }

    @Override
    public void deleteUser(String user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("deleteUser");
    }

    @Override
    public CommonUser updateUser(CommonUser user) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("updateUser");
    }

    @Override
    public void createRole(String role, boolean active) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("createRole");
    }

    @Override
    public void deleteRole(String role) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("deleteRole");
    }

    @Override
    public void updateRole(String role, boolean active) throws PrincipalAdapterExcept
        throw new UnsupportedOperationException("updateRole");
    }

    @Override
    public void assignRole(String user, String role) throws PrincipalAdapterException
        throw new UnsupportedOperationException("assignRole");
    }

    @Override
    public void removeRole(String user, String role) throws PrincipalAdapterException
        throw new UnsupportedOperationException("removeRole");
    }

    @Override
    public List<Profile> getProfiles(boolean filterByActive) throws PrincipalAdapterE
        throw new UnsupportedOperationException("getProfiles");
    }

    @Override
    public Profile getUserProfile(String userId) throws PrincipalAdapterException {
        throw new UnsupportedOperationException("getUserProfile");
    }

    @Override
    public void setUserProfile(String userId, long profileId) throws PrincipalAdapter
        throw new UnsupportedOperationException("setUserProfile");
    }
```

```
    @Override
    public boolean isManageUsers() {
        throw new UnsupportedOperationException("isManageUsers");
    }

    @Override
    public boolean isManageRoles() {
        throw new UnsupportedOperationException("isManageRoles");
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own parameters

You can take advantage of the application configuration parameters feature to store your own parameters.

To create a new configuration parameter:

- *Go to **Administration > Utilities > Database query**.*
- *Select **JDBC option** from the list at the bottom.*
- *Write a query like this*

```
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('your.property.nam
```

- *Click at **Execute button** at top right.*

*The **type value** - CFG_VALUE column - **must be written in lower case** !.*

Sample

You can access it from source code to the properties.

When the property not exists will be initialized with default value (on sample code below "initial value").

```
import com.openkm.db.service.ConfigSrv;
import com.openkm.util.ContextWrapper;

ConfigSrv configSrv = ContextWrapper.getContext().getBean(ConfigSrv.class);
// If the parameter not exists, by default is set to "initial value"
String value = configSrv.getString("your.property.name", "initial value");
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own password validator

You can create your own password validator.

Conditions:

- *The new password validator class **must implement the "PasswordValidator"** interface.*
- *The new password validator class **must be declared under the package "com.openkm.validator.password".***

Password validator interface:

```
package com.openkm.validator.password;

import com.openkm.validator.ValidatorException;

public interface PasswordValidator {

    /**
     * Validate
     *
     * @param password
     * @throws ValidatorException
     */
    void Validate(String password) throws ValidatorException;

}
```

From password validator class you can use several configuration parameters:

- *Config.PROPERTY_VALIDATOR_PASSWORD to return value of parameter "validator.password".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MIN_LENGTH to return value of parameter "validator.password.min.length".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MAX_LENGTH to return value of parameter "validator.password.max.length".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MIN_LOWERCASE to return value of parameter "validator.password.min.lowercase".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MIN_UPPERCASE to return value of parameter "validator.password.min.uppercase".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MIN_DIGITS to return value of parameter "validator.password.min.digits".*
- *Config.PROPERTY_VALIDATOR_PASSWORD_MIN_SPECIAL to return value of parameter "validator.password.mini.special".*

More information at Password validation configuration.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Sample

***CompletePasswordValidator** class:*

```
package com.openkm.validator.password;

import com.openkm.core.Config;
import com.openkm.validator.ValidatorException;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * Complex password validator
 */
public class CompletePasswordValidator implements PasswordValidator {

    @SuppressWarnings("unused")
    private static Logger log = LoggerFactory.getLogger(CompletePasswordValidator.cla

    @Override
    public void Validate(String password) throws ValidatorException {
        validateLength(password);
        checkLowerCase(password);
        checkUpperCase(password);
        checkDigits(password);
        checkSpecial(password);
    }

    /**
     * Validate length
     */
    private void validateLength(String password) throws ValidatorException {
        if (Config.VALIDATOR_PASSWORD_MIN_LENGTH > 0
                && password.length() < Config.VALIDATOR_PASSWORD_MIN_LENGTH) {
            throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MIN_LENGTH);
        }

        if (Config.VALIDATOR_PASSWORD_MAX_LENGTH > 0
                && password.length() > Config.VALIDATOR_PASSWORD_MAX_LENGTH) {
            throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MAX_LENGTH);
        }
    }

    /**
     * Validate lowercase characters
     */
    private void checkLowerCase(String password) throws ValidatorException {
        int count = 0;

        if (Config.VALIDATOR_PASSWORD_MIN_LOWERCASE > 0) {
            for (int i = 0; i < password.length(); i++) {
                if (Character.isLowerCase(password.charAt(i))) {
                    count++;
                }
            }

            if (Config.VALIDATOR_PASSWORD_MIN_LOWERCASE > count) {
                throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MIN_LOWE
            }
        }
    }

    /**
     * Validate uppercase characters
```

```
     */
    private void checkUpperCase(String password) throws ValidatorException {
        int count = 0;

        if (Config.VALIDATOR_PASSWORD_MIN_UPPERCASE > 0) {
            for (int i = 0; i < password.length(); i++) {
                if (Character.isUpperCase(password.charAt(i))) {
                    count++;
                }
            }

            if (Config.VALIDATOR_PASSWORD_MIN_UPPERCASE > count) {
                throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MIN_UPPE
            }
        }
    }

    /**
     * Validate digits
     */
    private void checkDigits(String password) throws ValidatorException {
        int count = 0;

        if (Config.VALIDATOR_PASSWORD_MIN_DIGITS > 0) {
            for (int i = 0; i < password.length(); i++) {
                if (Character.isDigit(password.charAt(i))) {
                    count++;
                }
            }

            if (Config.VALIDATOR_PASSWORD_MIN_DIGITS > count) {
                throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MIN_DIGI
            }
        }
    }

    /**
     * Validate special characters
     */
    private void checkSpecial(String password) throws ValidatorException {
        int count = 0;

        if (Config.VALIDATOR_PASSWORD_MIN_SPECIAL > 0) {
            for (int i = 0; i < password.length(); i++) {
                if (!Character.isLetterOrDigit(password.charAt(i))
                        && !Character.isWhitespace(password.charAt(i))) {
                    count++;
                }
            }

            if (Config.VALIDATOR_PASSWORD_MIN_SPECIAL > count) {
                throw new ValidatorException(Config.VALIDATOR_PASSWORD_ERROR_MIN_SPEC
            }
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own reports

OpenKM use JasperReports library for generating reports. For building reports you have a tool named iReport.

*The **JasperReports Library** is the world's most popular Open Source reporting engine. It is entirely written in Java and it is able to use data coming from any kind of data source and produce pixel-perfect documents that can be viewed, printed or exported in a variety of document formats including HTML, PDF, Excel, OpenOffice and Word. More information at http://community.jaspersoft.com/.*

***iReport** is the free, Open Source report designer for JasperReports and JasperReports Server. Create very sophisticated layouts containing charts, images, subreports, crosstabs and much more. Access your data through JDBC, TableModels, JavaBeans, XML, Hibernate, CSV, and custom sources. Then publish your reports as PDF, RTF, XML, XLS, CSV, HTML, XHTML, text, DOCX, or OpenOffice. More information at http://community.jaspersoft.com/project/ireport-designer.*

In this section you will find an introduction for creating reports with JasperReports and iReports tool. Consider it as a quick introduction, not a complete guide. You must follow some tutorials, read books or papers to illustrate on it.

There are some available books and guides you should read:

- *The Definitive Guide to iReport by Giulio Toffoli.*
- *JasperReports Ultimate Guide.*
- *Ireport Designer Tutorials & Help.*

With OpenKM you can use reports based on SQL or reports based on Scripting. For beginners we suggest starting with reports based on SQL.

- *Introducing to reports based on SQL.*
- *Introducing to reports based on Java.*
- *Introducing to reports with filtering.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Introducing to reports based on SQL

- *Install **iReport** tool.*
- ***Execute iReport** tool.*

OpenKM comes with an specific JasperReports Library engine what supports older version but now newer. At Application version compatibility table you have correspondence list between OpenKM and JasperReports Library engine versions.

You can set specific output version from iReport tool:

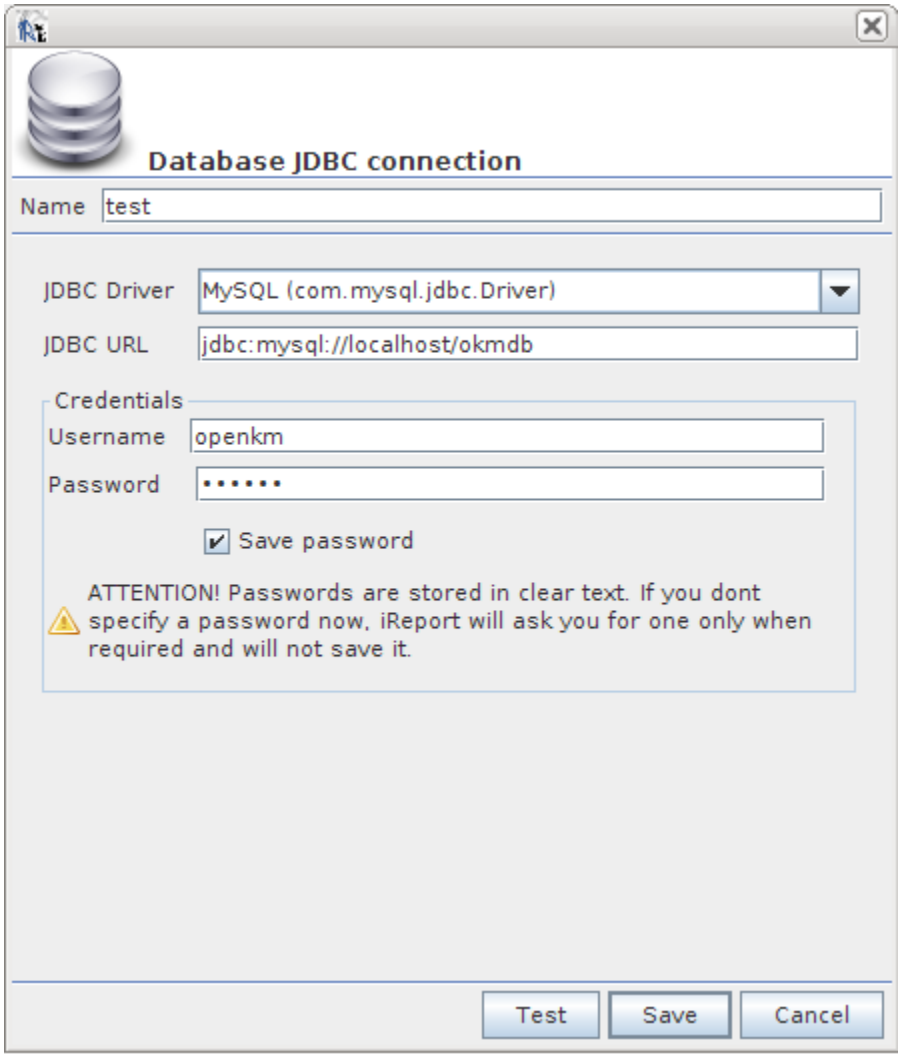
- *Click on **Tools menu > options**.*
- *From **iReport > General tab** click on **Compatibility tab**.*
- *Set the **Jasper version compatibility**.*
- *Click on **OK button**.*

Create datasource

- *Click on **Report datasources button**.*

- *Click on **New button**.*
- *Choose "**Database JDBC connection**" option.*
- *Click on **Next button**.*
- *Set the **name**.*
- *Choose the **JDBC driver**.*
- *Click on **Text button** to check the connection parameters.*
- *Click on **Save button**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



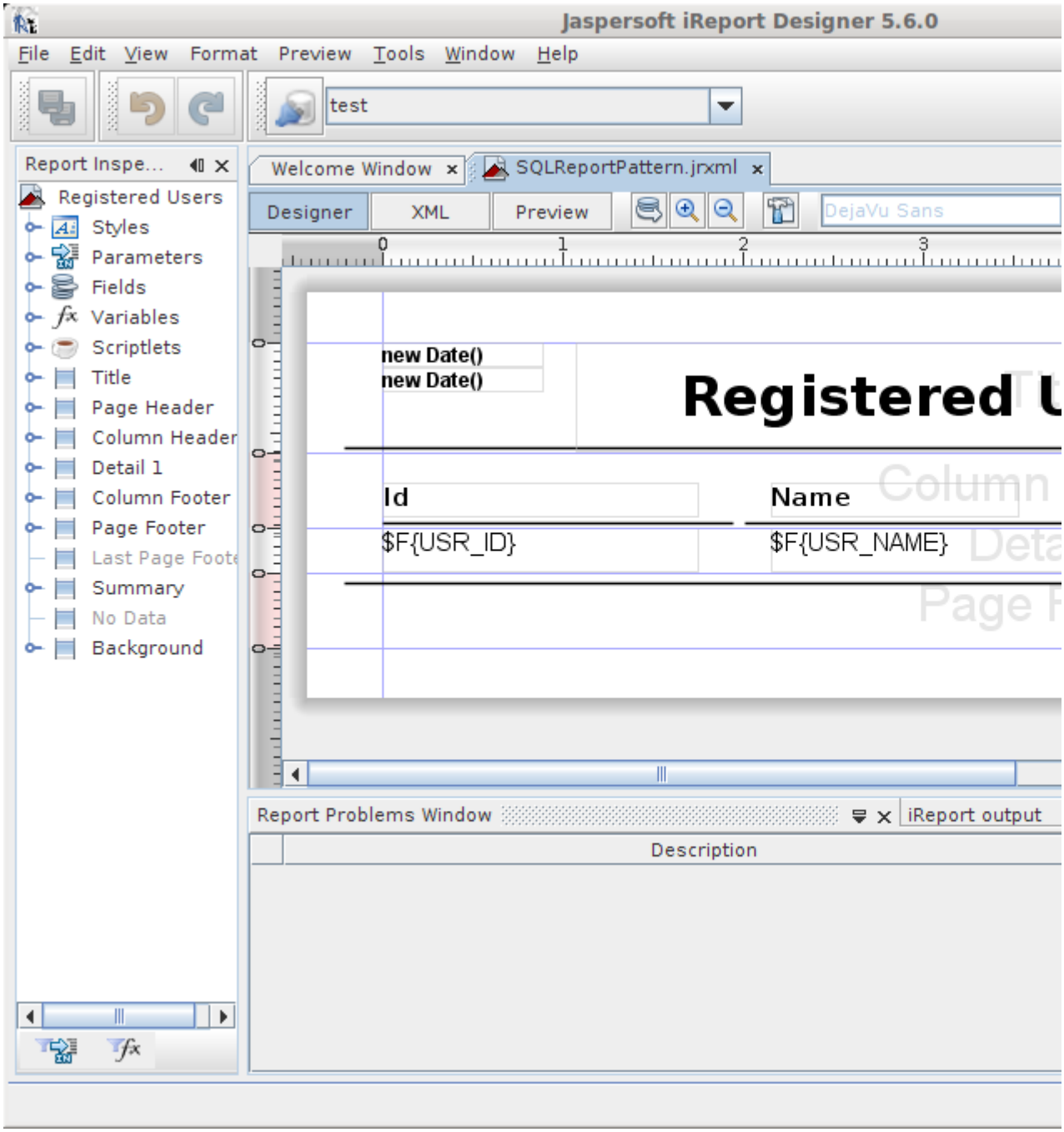
- ***Check as default checkbox***.
- *Click on* ***Close button***.

Testing SQL sample report

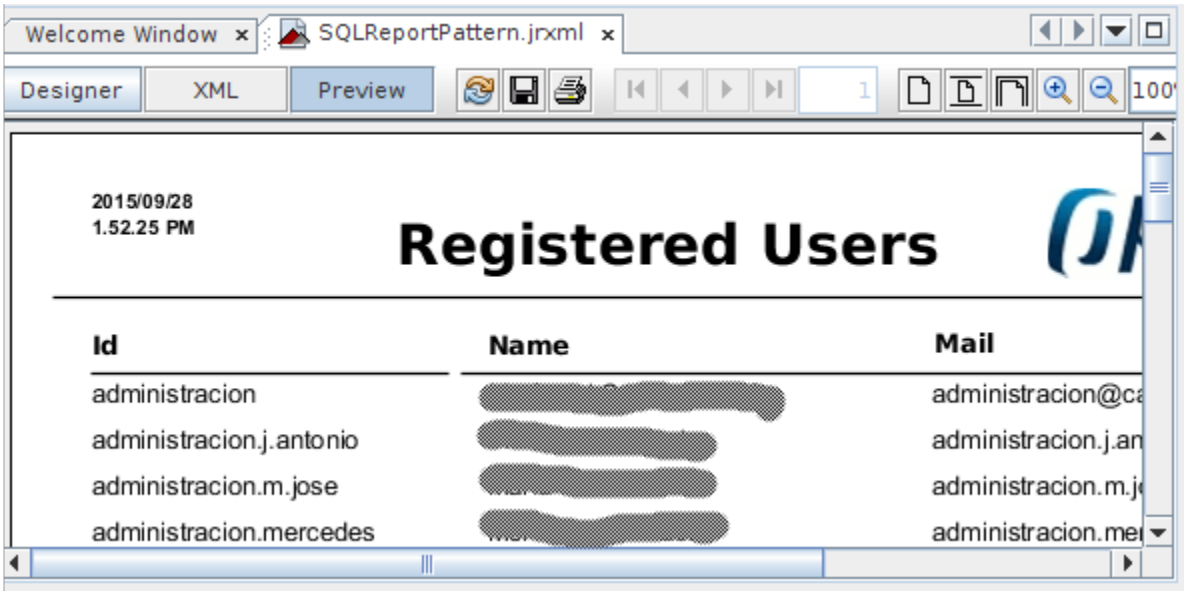
> *For testing the SQL Report sample you must have an OpenKM running.*

- *Download the SQLReportPattern.jrxml file.*
- *Open the file.*

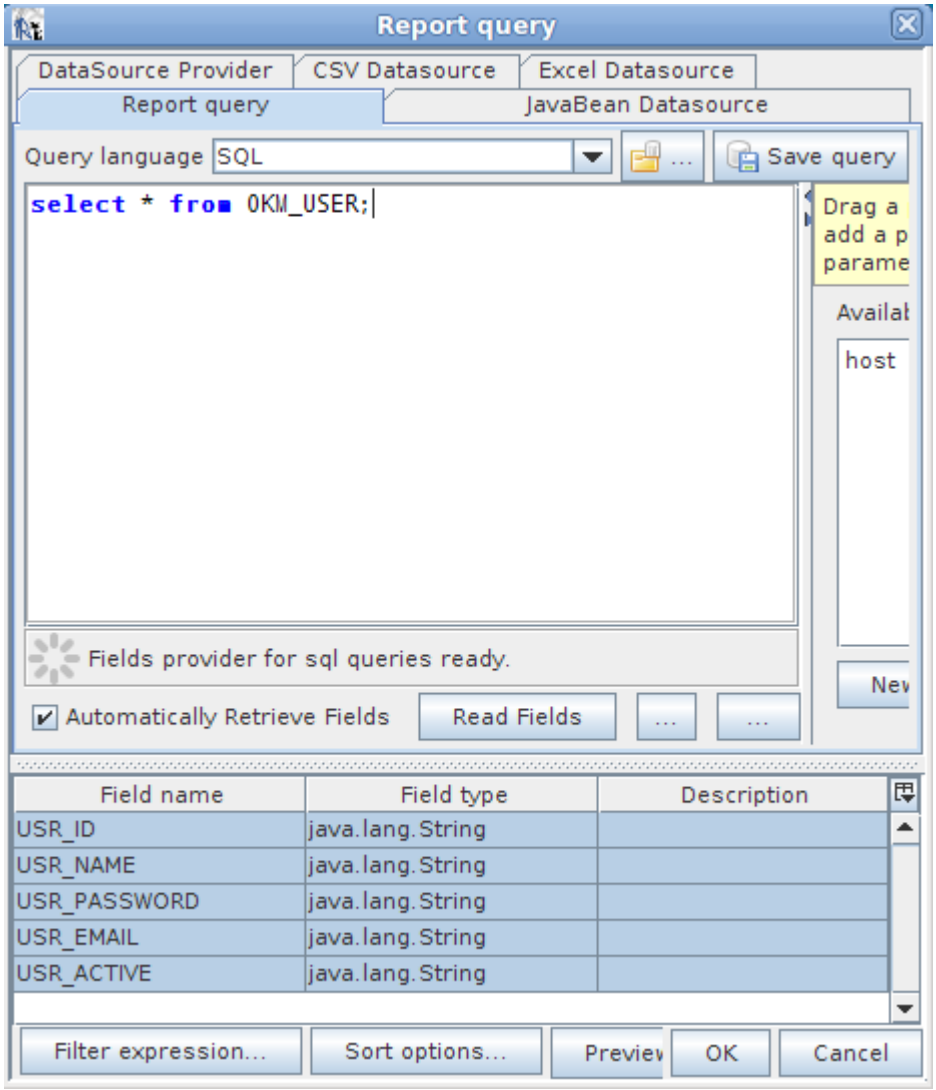
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



- *Click on **Preview tab**.*
- *Set the **host** value (for example http://localhost:8080/OpenKM).*



- *Click on* ***Designer tab***.
- *Click on* ***report query button***.



From this point you can add your jrxml file and register to your OpenKM at ***Administration*** > *Reports*.

Closer look into JRXML file

The iReport tool is creating for you the jasper report file in XML format.

Although the iReport tool is doing all work for you in friendly user interface, is a good idea for you to take a look - at least for curiousity - about what kind of file is being created.

Query section

```
<queryString>
  <![CDATA[select * from OKM_USER;]]>
</queryString>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Fields section

```
<field name="USR_ID" class="java.lang.String"/>
<field name="USR_NAME" class="java.lang.String"/>
<field name="USR_PASSWORD" class="java.lang.String"/>
<field name="USR_EMAIL" class="java.lang.String"/>
<field name="USR_ACTIVE" class="java.lang.String"/>
```

Remote image section

```
<imageExpression><![CDATA[$P{host}+"/logo/report"]]></imageExpression>
```

Using fields in report

```
<textField isBlankWhenNull="false">
  <reportElement key="textField" x="329" y="0" width="206" height="18" uuid="5e3e2df8
    <textElement>
      <font fontName="Arial"/>
    </textElement>
  <textFieldExpression><![CDATA[$F{USR_EMAIL}]]></textFieldExpression>
</textField>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Introducing to reports based on Java

- *Install the **iReport** tool.*
- ***Execute iReport** tool.*

OpenKM comes with an specific JasperReports Library engine that supports older versions but now a newer one can be found. At Application version compatibility table you have correspondence list between OpenKM and JasperReports Library engine versions.

You can set specific output version from iReport tool:

- *Click on **Tools menu > options**.*
- *From **iReport > General tab** click on **Compatibility tab**.*
- *Set the **Jasper version compatibility**.*
- *Click on **OK button**.*

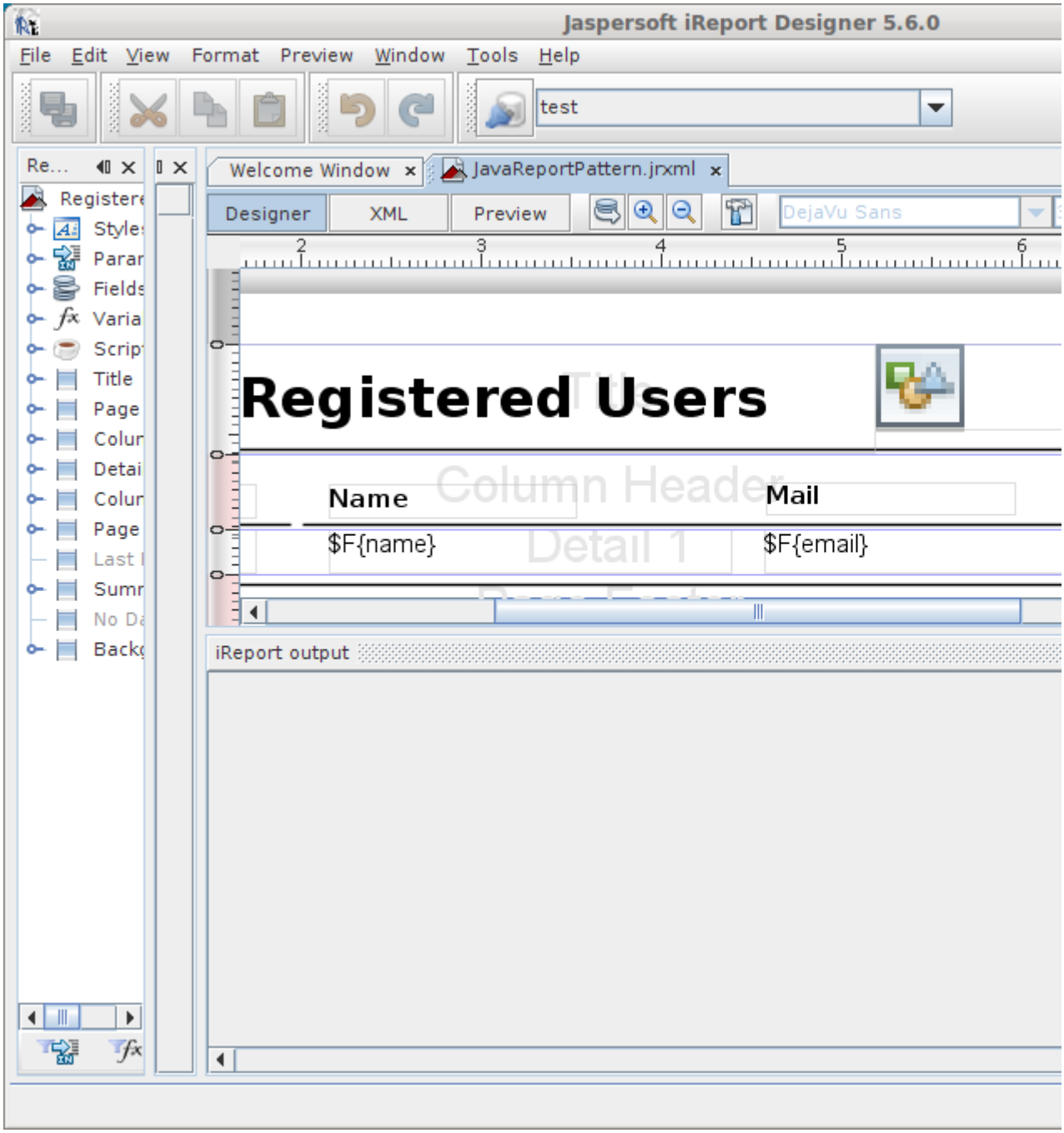
Testing Java sample report

***Reports based on Java can not be previewed directly from iReport tool.** Must be checked from an OpenKM.*

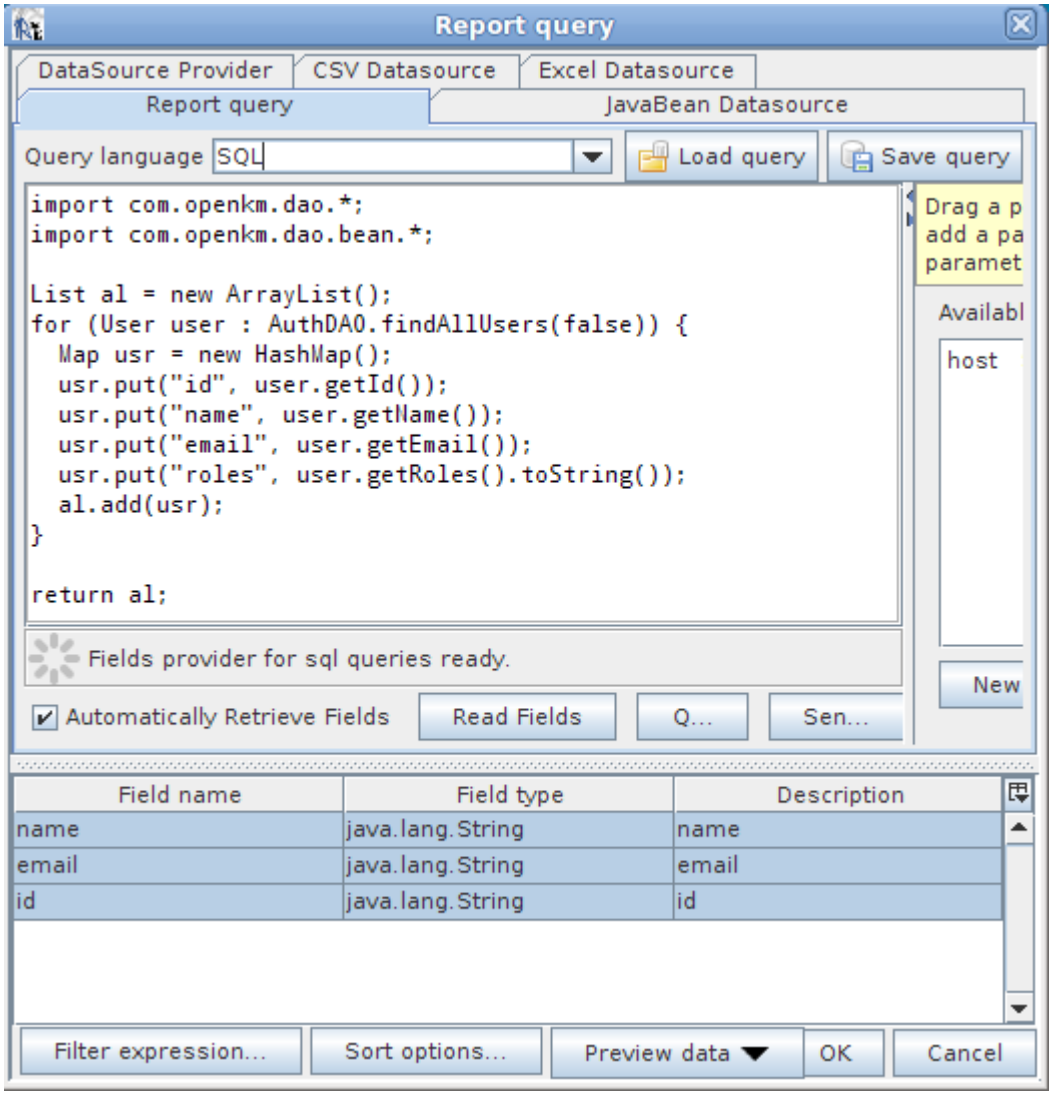
For testing SQL Report sample you must have an OpenKM running.

- *Download the JavaReportPattern.jrxml file.*
- *Open the file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



- *Click on* ***Designer tab***.
- *Click on* ***report query button***.



In this case in place of SQL query you can set a Java Scripting (see also Scripting samples).

The script must return a List<Map> object where each value into the map corresponds to a column value.

From this point you can add your jrxml file and register to your OpenKM at ***Administration*** *>* Reports.

Understanding the code sample

- *Create an empty list.*
- *Iterate across all users calling the API method named AuthDAO.findAllUsers(false).*
- *Create a Map for each User object where each value corresponds to a column.*
- *Add each map into the list, where each map corresponds to a row.*

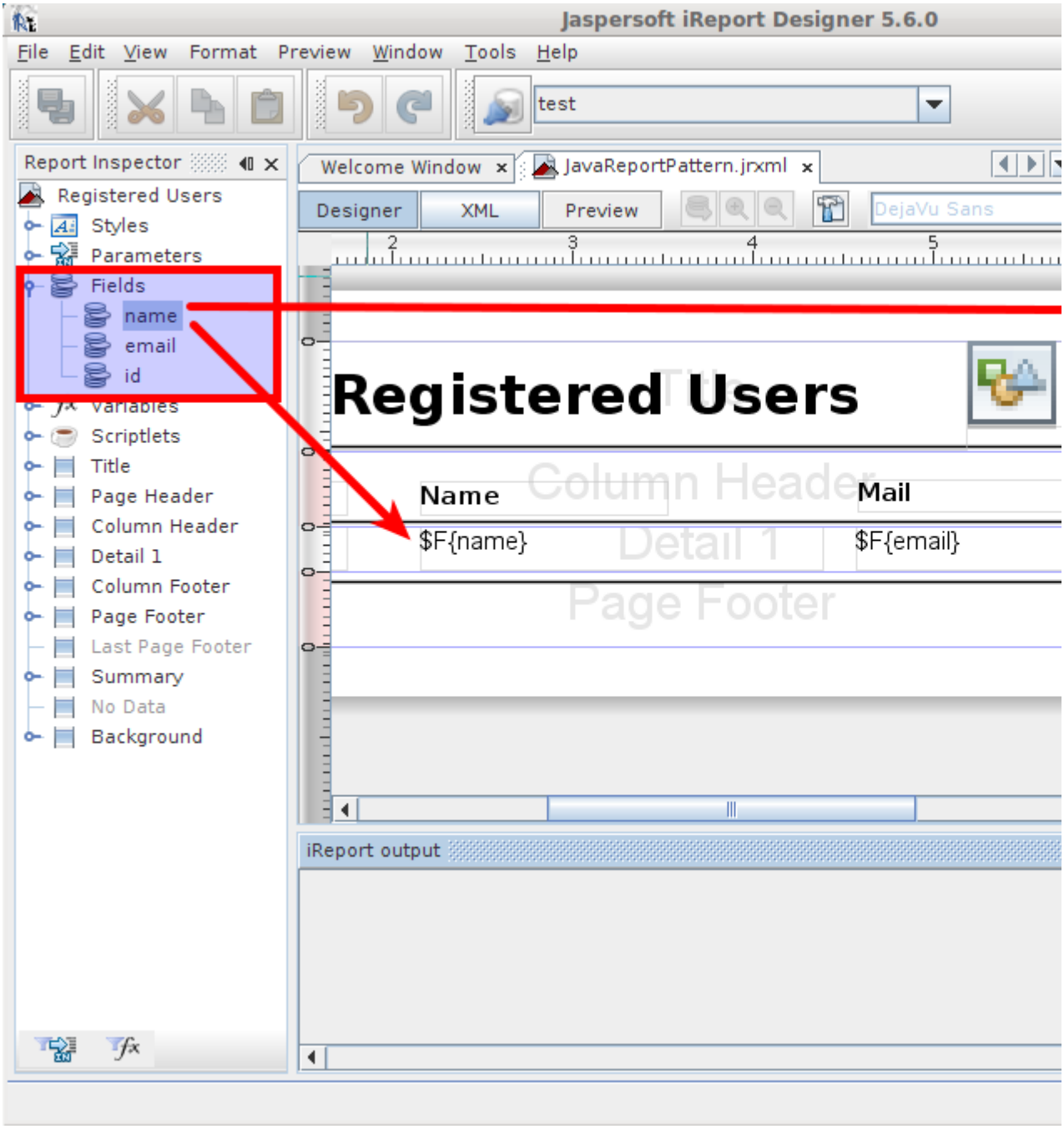
```
import com.openkm.dao.*;
import com.openkm.dao.bean.*;
import com.openkm.util.ContextWrapper;
import com.openkm.db.service.AuthSrv;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
AuthSrv authSrv = ContextWrapper.getContext().getBean(AuthSrv.class);

List al = new ArrayList();
for (DbUser user : authSrv.findAllUsers(false)) {
    Map usr = new HashMap();
    usr.put("id", user.getId());
    usr.put("name", user.getName());
    usr.put("email", user.getEmail());
    usr.put("roles", user.getRoles().toString());
    al.add(usr);
}

return al;
```

Using in report

- *For each key in map create a field.*
- *Use the field in report*



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Introducing to reports with filtering

When you execute a report from OpenKM the view might be shown a form requiring the user to introduce values for filtering purpose. The most common case is filtering by a range of dates.

*The reports with form filtering ends with extension "**.rep**". Really is a ZIP file renamed as "**.rep**".*

These reports contains into two files:

- *JRXML file*
- *File named params.xml*

Download the UsersLogin.zip and take a look into (this is the original UsersLogin.rep renamed to ZIP).

Understanding filtering

For understanding form filter we will use the UsersLogin.rep report (dowload the UsersLogin.zip).

Params XML file

In the sample has been set two filtering fields:

Field	*Type*	*Description*
from_date	***Input***	*Field used to filtering from date.*
to_date	***Input***	*Field used to filtering to date.*

The params.xml file:

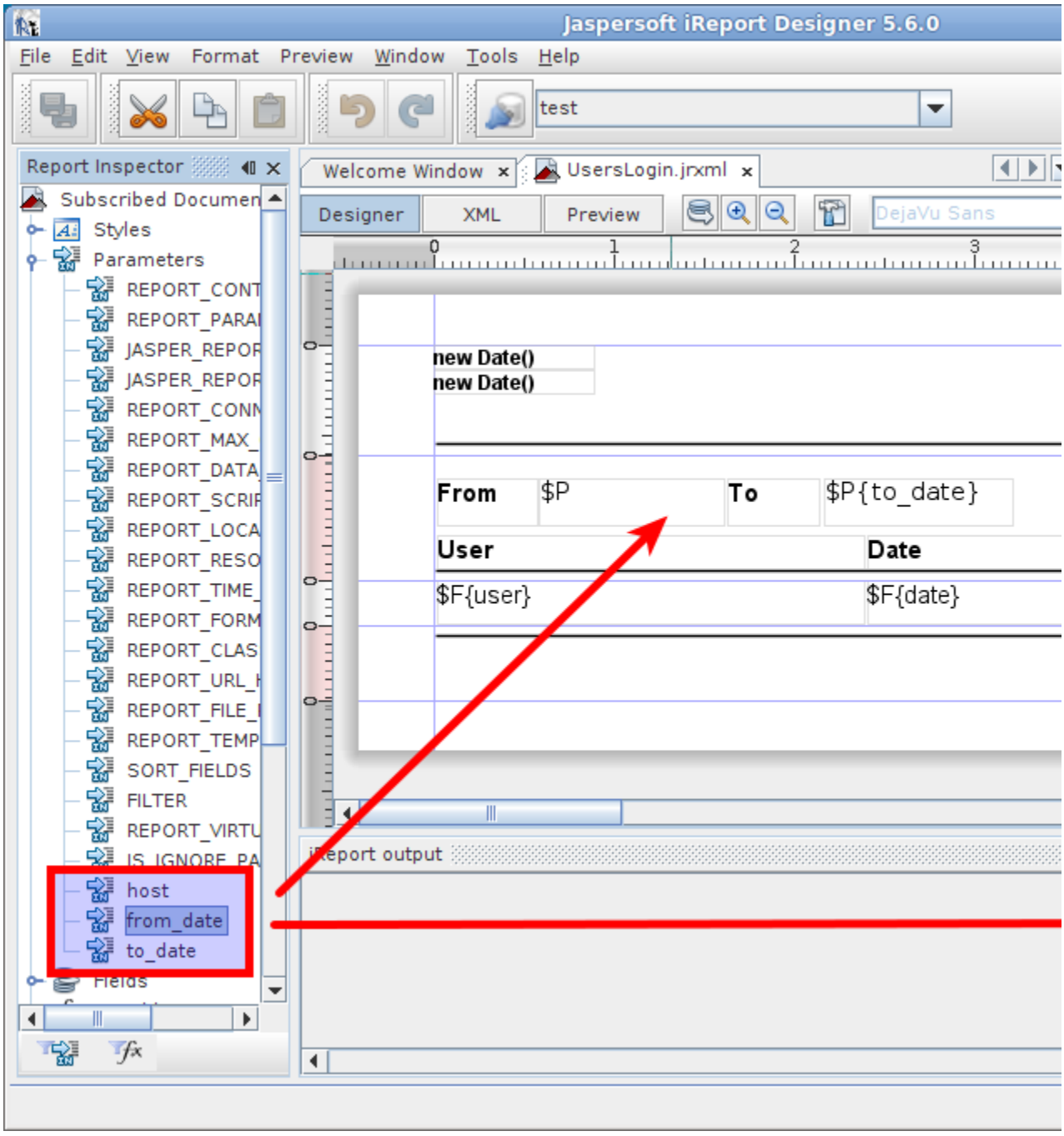
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.0//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.0.d
<report-parameters>
  <input label="From" name="from_date" type="date">
    <validator type="req"/>
  </input>
  <input label="To" name="to_date" type="date">
    <validator type="req"/>
  </input>
</report-parameters>
```

Using fields in report

In the report are created two parameters:

Parameter	*Class*	*Description*

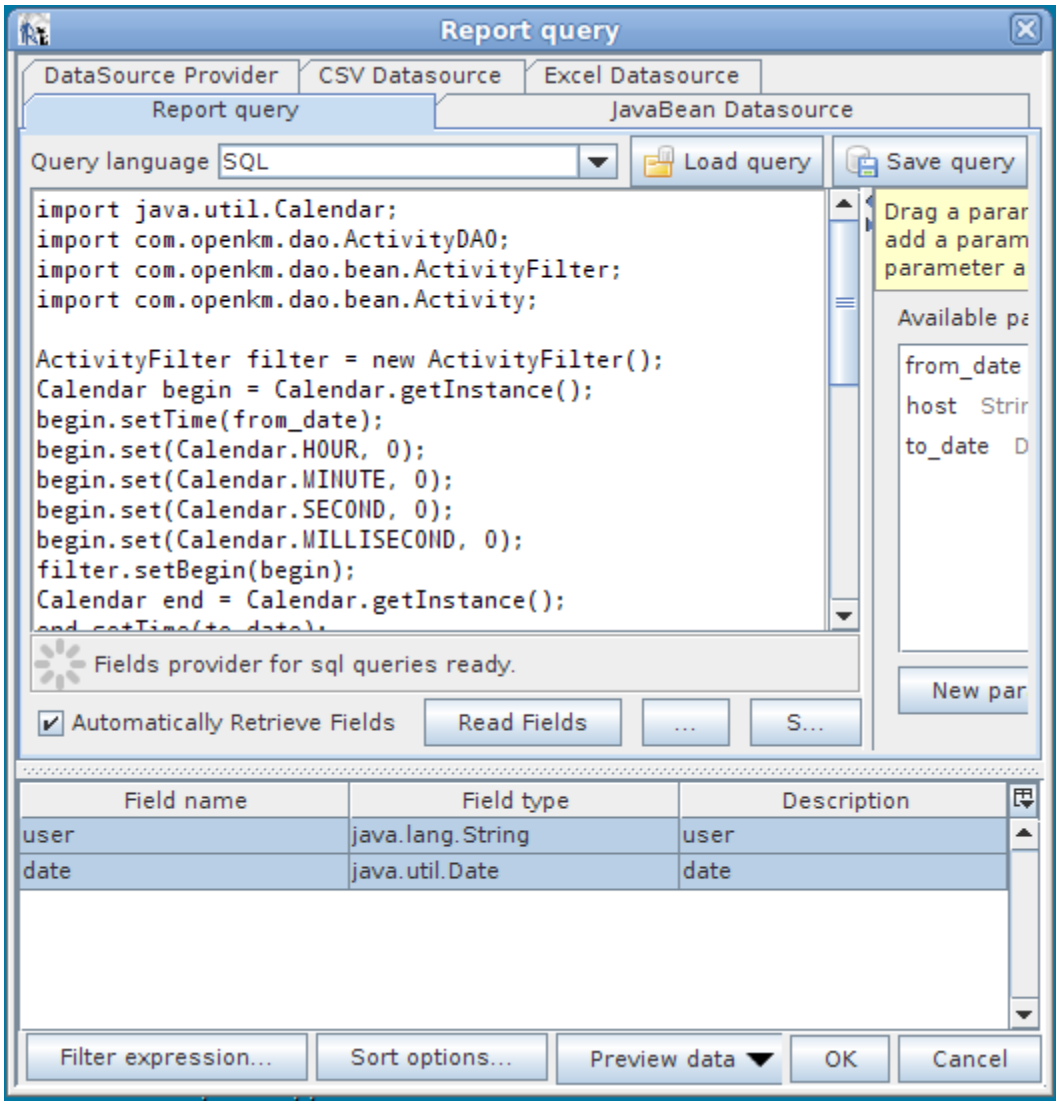
from_date	***java.util.Date***	*From date.*
to_date	***java.util.Date***	*To date.*



The parameters are also used in SQL or Java reports.

Java sample

- *Click on* ***report query button***.



The ***from_date*** *and* ***to_date*** *variables are injected in code execution and can be used directly as sample below.*

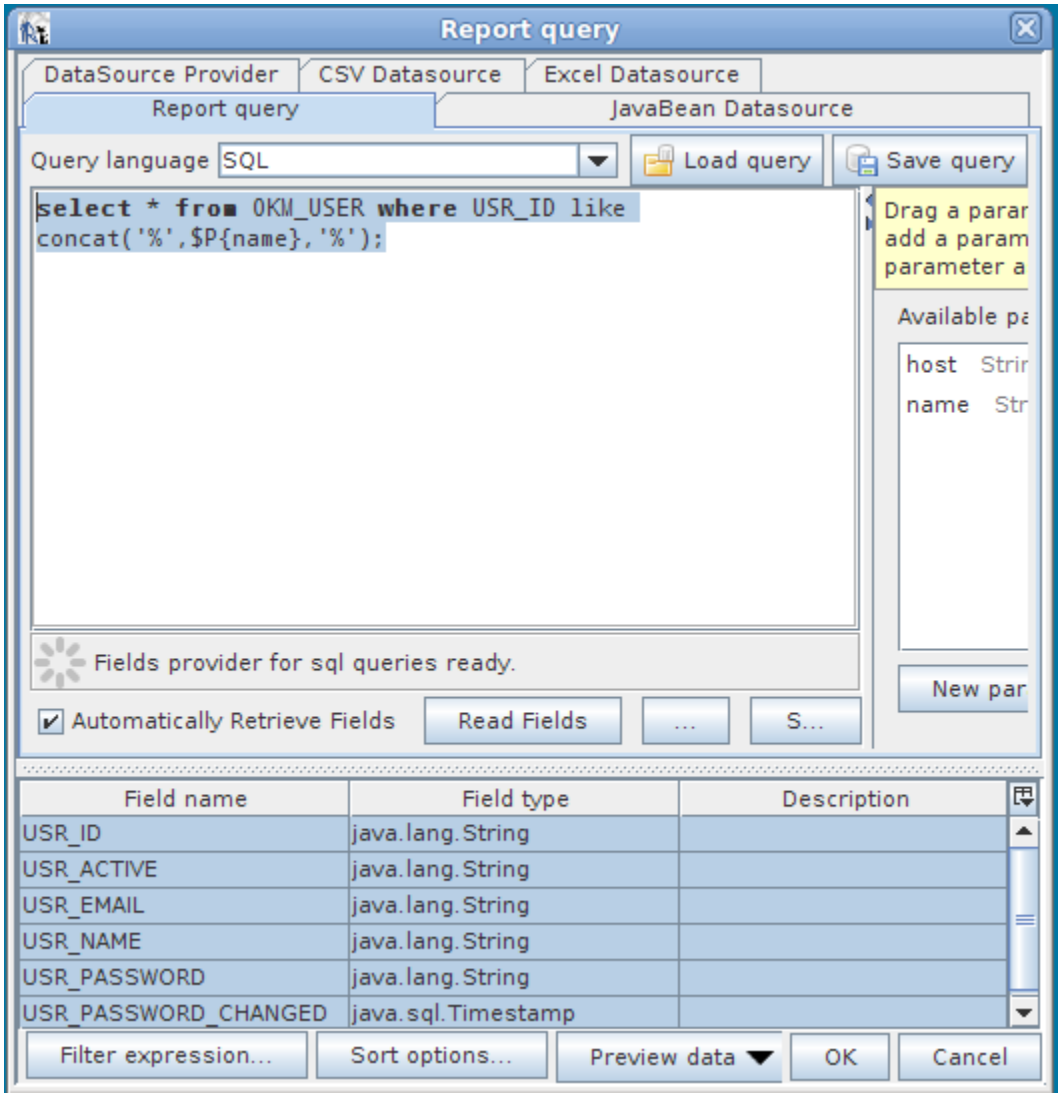
```
import java.util.Calendar;
import com.openkm.dao.ActivityDAO;
import com.openkm.dao.bean.ActivityFilter;
import com.openkm.dao.bean.Activity;

ActivityFilter filter = new ActivityFilter();
Calendar begin = Calendar.getInstance();
begin.setTime(from_date);
begin.set(Calendar.HOUR, 0);
begin.set(Calendar.MINUTE, 0);
begin.set(Calendar.SECOND, 0);
begin.set(Calendar.MILLISECOND, 0);
filter.setBegin(begin);
Calendar end = Calendar.getInstance();
end.setTime(to_date);
end.set(Calendar.HOUR, 0);
end.set(Calendar.MINUTE, 0);
```

```
end.set(Calendar.SECOND, 0);
end.set(Calendar.MILLISECOND, 0);
filter.setEnd(end);
filter.setAction("LOGIN");
List al = new ArrayList();

for (Iterator it = ActivityDAO.findByFilter(filter).iterator(); it.hasNext();){
    Activity act = (Activity) it.next();
    Map actMap = new HashMap();
    actMap.put("user", act.getUser());
    actMap.put("date", act.getDate().getTime());
    al.add(actMap);
}

return al
```

SQL sample

The sample below is not based on the UsersLogin.rep report.

Click on ***report query button***.



*In the sample there is a parameter named "**name**" of class "**java.lang.String**" used for filtering users list.*

```
select * from OKM_USER where USR_ID like concat('%',$P{name},'%')
```

The engine replaces "$P{name}" tag to "String" value. For example if parameter value is "okm" the query executed will be:

```
select * from OKM_USER where USR_ID like concat('%','okm','%')
```

The contact function merges several Strings, the final result is:

```
select * from OKM_USER where USR_ID like '%okm%')
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params form fields

Params forms are based on formal XML definition. The DTD (Document Type Definition) defines the structure and the legal elements and attributes of an XML document.

> *Can see latest DTD in our website http://www.openkm.com/dtd/report-parameters-{major}.{minor}.dtd where **{major}** and **{minor}** values are two numbers (major release and minor release). For example the location for version 2.1 of the dtd is http://www.openkm.com/dtd/report-parameters-2.1.dtd*

> *Each application version supports only a couple of DTD definitions, read Application version compatibility table to see which are supported by your application.*

Basic XML skeleton

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>

</report-parameters>
```

- *The **DOCTYPE** is a formal definition of some DTD. In the previous example the **report-parameters-2.1.dtd**.*
- *The XML begins with **<report-parameters> tag** and ends with **</report-parameters>**. Into this hicheracy are defined the metada groups.*

Basic example

Available field types:

- *Checkbox*
- *Input*
- *Select*
- *Separator*
- *Suggestbox*
- *Text*
- *TextArea*

More information about field type elements at Params fields

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <input label="From" name="from_date" type="date">
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    <validator type="req"/>
  </input>
  <input label="To" name="to_date" type="date">
    <validator type="req"/>
  </input>
</report-parameters>
```

Using file system DTD

If the server has no access to the Internet, then the DTD file must be accessible in some way, for example can store a copy in your application server.

The XML definition must be changed to refer the new DTD location, otherwise you will get and application error while trying to processing DTD file.

Can be file system (example below) or a valid intranet URL.

```
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.1//EN"
                                 "file:///home/openkm/report-parameters-2.1.dtd">
```

- *In the example, **/home/openkm/report-parameters-2.1.dtd** is the file system path to the file.*

Params fields

These are the available field types:

Field type	*Description*
Checkbox	*Params Checkbox field*
Input	*Params Input field*
Select	*Params Select field*
Separator	*Params Separator field*
Suggestbox	*Params Suggestbox field*
Text	*Params Text field*
TextArea	*Params TextArea field*

Some field types can have children elements, validator or option elements. This hierarchy is permitted:

- *params form*
  - *Checkbox*
    - *Validator*
  - *Input*
    - *Validator*
  - *Select*
    - *Option*
    - *Validator*
  - *Separator*
  - *Suggestbox*
    - *Validator*
  - *Text*
  - *TextArea*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Validator*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Validator element

The goal of the Validator element is to ensure that the user provided necessary and properly formatted information needed to successfully complete an operation.

Validator can only be used in combination with other fields:

- *Params Input field.*
- *Params Suggestbox field.*
- *Params Checkbox field.*
- *Params TextArea field.*
- *Params Select field.*

Available validators:

name	*Description*
req	*Required value.*
alpha	*Alphanumeric value.*
dec	*Decimal value.*
num	*Numeric value.*
email	*E-mail value.*
url	*URL value.*
maxlen	*Set the maximum length allowed.*
minlen	*Set the minimum length required.*
lt	*Less than.*
gt	*Greater than.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

min	*Minimum value.*
max	*Maximum value.*
regexp	*Regular expression.*
plugin	*Create your own form validator.* *More information at Creating your own Form Validator plugin.*

Parent:

- *Params Input field.*
- *Params Suggestbox field.*
- *Params Checkbox field.*
- *Params TextArea field.*
- *Params Select field.*

Basic example

- *Value required "**req**".*
- *Value must be decimal "**dec**".*
- *The maximum lenght of the decimal value is 6 digits "**maxlen**".*

```
<input label="Decimal value" name="decimal">
    <validator type="dec"/>
    <validator type="req"/>
    <validator type="maxlen" parameter="6"/>
</input>
```

Regular expression

If you want a validation which is not included in these prefefined, you can use a regular expression. For example, imagine you want to restrict an input to lowecase ASCII values:

```
<input label="Lowercase value" name="regExp">
    <validator type="regexp" parameter="[a-z]+" />
</input>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Input field

in most cases is used to enter free text form, although can be used to save http links, date data or refer to application folder (as internal link to this folder).

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier.* *Two fields can't have the same name into the same task definion.* ***Name must be unique****.*	*true*
type	*Type value. Allowed types:* • *text.* • *date.* • *link.* • *folder.*	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

Child elements:

- *Validator. For more information see Params Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <input label="From" name="from_date" type="date">
    <validator type="req" />
  </input>
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Suggestbox field

This is an implementation of the typical AJAX suggest box component:

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier.* ⚠ *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
table	*Metadata table name.* *Application can store values into metadata table, that can be used in this query.*	*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
filterQuery	*An Hibernate query to filter the data. Use {0} to place the user input.*	*
valueQuery	*An Hibernate query to obtain the value of a identifier. Use {0} to place the identifier.*	*
className	*A class used to get the selected options.*	*
filterMinLen	*The minimum chars in input to begin to filter results.*	*true*
dialogTitle	*The title of the HTML dialog.*	*true*

Child elements:

- *Validator. For more information see Params Validator element.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

** There're three options to get suggestbox values (**use one of them is mandatory**):*

- *table + filterQuery + valueQuery*
- *filterQuery + valueQuery*
- *className*

No other combination of them is allowed.

Based on table, filterQuery and valueQuery

Suggestbox values are defined in metadata tables.

- *Application use two tables named OKM_DB_METADATA_TYPE AND OKM_DB_METADATA_VALUE that helps on storing options values used in selects. That's very useful when you got a large number of values.*
- *In table OKM_DB_METADATA_TYPE are stored metadata table definition (table name, column name and column type).*
- *In table OKM_DB_METADATA_VALUE are stored record values.*

Create metadata table definition.

*The **table** will be called **country**, with two columns, **country_id** and **country_name**.*

*The **country_id** column is **integer** type and **country_name** is **text** type.*

*The **country_id** is set as **column 0 with col00** value.*

*The **country_name** is set as **column 1 with col01** value.*

- *Go to **Administration > Utilities > Database query**.*
- *At bottom right select **jdbc** from list.*
- ***Paste the SQL** in the box*

```
DELETE FROM OKM_DB_METADATA_TYPE WHERE DMT_TABLE='country';
INSERT INTO OKM_DB_METADATA_TYPE (DMT_TABLE, DMT_REAL_COLUMN, DMT_TYPE, DMT_VIRTUAL_C
INSERT INTO OKM_DB_METADATA_TYPE (DMT_TABLE, DMT_REAL_COLUMN, DMT_TYPE, DMT_VIRTUAL_C
```

- *Click at bottom right the **Execute button**.*

Insert metadata table values:

- *Go to **Administration > Utilities > Database query**.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *At bottom right select **jdbc** from list.*
- ***Paste the SQL** in the box*

```
DELETE FROM OKM_DB_METADATA_VALUE WHERE DMV_TABLE='country';
INSERT INTO OKM_DB_METADATA_VALUE (DMV_TABLE, DMV_COL00, DMV_COL01) VALUES ('country'
INSERT INTO OKM_DB_METADATA_VALUE (DMV_TABLE, DMV_COL00, DMV_COL01) VALUES ('country'
```

- *Click at bottom right the **Execute button**.*

The insert queries depending your database can not be match, for example in Oracle:

*INSERT INTO OKM_DB_METADATA_TYPE (**DMT_ID**, DMT_TABLE, DMT_REAL_COLUMN, DMT_TYPE, DMT_VIRTUAL_COLUMN) VALUES (**HIBERNATE_SEQUENCE.NEXTVAL**, 'country', 'col00', 'integer', 'country_id');*

Finally the XML definition:

- *In optionsQuery the values that begins with character $ are identified as column names. See **$country_id and $country_name** below.*
- *Can use normal SQL clausules like "**lower**", "**like**", "**order by**", etc.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
    <suggestbox label="country" name="suggestbox" width="200px"
                table="country" dialogTitle="Choose country" filterMinLen="3"
                filterQuery="select $country_id, $country_name from DatabaseMetad
                valueQuery="select $country_id, $country_name from DatabaseMetada
</report-parameters>
```

Based on filterQuery and valueQuery (without table attribute)

*Now **filterQuery** and **valueQuery** could be a regular SQL query.*

You should only use {0} as the parameter entered by the user to filter the values from the table.

Create COUNTRY table:

```
create table COUNTRY (
  CT_ID varchar(2),
  CT_NAME varchar(32),
  primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
    <suggestbox label="country" name="okp:consulting.suggestbox"
                    width="200px" dialogTitle="Choose country" filterMinLen="3"
                    filterQuery="select ct_id, ct_name from country where ct_name li
                    valueQuery="select ct_id, ct_name from country where ct_id='{0}'
</report-parameters>
```

Based on className

To help getting a list from external applications or other purposes, your own class can be built to retrieve the option values from those systems .

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
    <suggestbox label="country" name="okp:consulting.suggestbox"
                    width="200px" dialogTitle="Choose country" filterMinLen="3"
                    className="com.openkm.plugin.form.values.SuggestBoxUserList" />
</report-parameters>
```

For more information read Creating your own Suggestbox plugin

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Checkbox field

Checkbox

Best used to represent boolean data:

Attribute	Description	Required
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

Child elements:

- *Validator. For more information see Params Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <checkbox label="Checkbox label" name="checkbox" />
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params TextArea field

Print a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task defition.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

Child elements:

- *Validator. For more information see Params Validator element.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <textarea label="textarea label" name="textarea" width="200px" />
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Select field

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier.* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
type	*Type value. Allowed types:* • *simple (single choice allowed).* • *multiple (multiple choice allowed).*	*true*
table	*Metadata table name.* *Application can store values into metadata table, that can be used in this query.*	*****
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*
optionsQuery	*A metadata query or SQL query used to get the select options.*	*****

Child elements:

- *Option. Couples of XML* ***<option label="label text" value="unique identifier value"/>*** *values or* ***<option label="label text" value="unique identifier value" parentValue="parent unique identifier value"/>*** *in case is set some hierarchical relationship.*
- *Validator. For more information see Params Validator element.*

***** *There're two options to get option values:*

- *Directly defined in XML.*
- *Metadata table (table + optionsQuery).*

One of these options must be defined, otherwise you will get an error. A combination of several of these cannot be done , only one can be choosen.

Basic select example

- *Type is equal to **simple**.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <select label="select label" name="select" type="simple">
    <option label="one" value="001" />
    <option label="two" value="002" />
    <option label="three" value="003" />
  </select>
</report-parameters>
```

Allow multiple choice

- *Type is equal to **multiple**.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <select label="select label" name="select" type="multiple">
    <option label="one" value="001" />
    <option label="two" value="002" />
    <option label="three" value="003" />
  </select>
</report-parameters>
```

Hierarchical relationship

- *The field option values of "**childrenSelect**" have parent relation with "**parentSelect**". It's defined by **parentElement="parentSelect"**.*
- *The option value "**benefits**" of field "**childrenSelect**" has a parent value "**directives**" from field "**parentSelect**".*

Tree relationships:

- *directives*
  - *benefits*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *eligibility*
- *services*

- *handbook*
  - *basic*
  - *normal*
  - *advanced*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <select label="Parent" name="parentSelect" type="simple">
    <option value="directives" label="Directives" />
    <option value="handbook" label="Handbook Paragraphs" />
  </select>
  <select label="Children" name="childrenSelect" type="simple" parentElement="parentS
    <option value="benefits" label="Benefits" parentValue="directives" />
    <option value="elegibility" label="Eligibility" parentValue="directives" />
    <option value="services" label="Services" parentValue="directives" />
    <option value="basic" label="Basic" parentValue="handbook" />
    <option value="normal" label="Advanced" parentValue="handbook" />
    <option value="advanced" label="Normal" parentValue="handbook" />
  </select>
</report-parameters
```

Based on SQL query

We can also get these option values using an SQL query. This SQL sentence should be put on the ***optionsQuery*** *attribute, but the* ***table*** *attribute should be empty (otherwise this query is used as a metadata query). Note that the first element returned in every row is the option value and the second one is the option label.*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <select label="country" name="country" type="simple"
          optionsQuery="select CT_ID, CT_NAME from COUNTRY order by CT_NAME" />
</report-parameters>
```

The table used in this sample is defined as:

```
create table COUNTRY (
    CT_ID varchar(2),
    CT_NAME varchar(32),
    primary key(CT_ID)
);

insert into COUNTRY (CT_ID, CT_NAME) values ('es', 'Spain');
insert into COUNTRY (CT_ID, CT_NAME) values ('pt', 'Portugal');
insert into COUNTRY (CT_ID, CT_NAME) values ('it', 'Italy');
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Based on className

To help getting a list from external applications or other purposes, your own class can be built to retrieve the option values from those systems .

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
    <select label="select label" name="okp:consulting.select" type="simple"
            className="com.openkm.plugin.form.OptionSelectUserList" />
</report-parameters>
```

For more information read Creating your own Option Select Values plugin.

Suggested options

Based on the content analyzer the application can suggest some options values. At the user interface the suggested options will be shown in green.

The application has some implementations that can easily be extended.

- ***com.openkm.form.suggestion.DocumentContentContainsSuggestion**: Get suggestions based on document content (use String.contains)*
- ***com.openkm.form.suggestion.DocumentContentTokenizerSuggestion**: Get suggestions based on document content (Use StringTokenizer).*
- ***com.openkm.form.suggestion.DocumentTermsSuggestion**: Get suggestions based on document term vectors.*

For more information read Creating your own Suggestion Analyzer.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
    <select label="select label"  name="okp:consulting.select" type="multiple"
            suggestion="com.openkm.form.suggestion.DocumentContentContainsSuggestion"
        <option label="one" value="001" />
        <option label="two" value="002" />
        <option label="three" value="003" />
    </select>
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Separator field

Draw a separator between fields.

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique****.*	*true*
width	*The width of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <input label="Input label" name="input1" />
  <separator label="Separator label" name="separator" />
  <input label="Input label" name="input2" />
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Params Text field

Print a text that can contain HTML tags.

Attribute	*Description*	*Required*
label	*The text shown as label in user interface.*	*true*
name	*Unique field identifier* *Two fields can't have the same name in the same task definion.* ***Name must be unique***.	*true*
width	*The width of the HTML element.*	*false*
height	*The height of the HTML element.*	*false*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report-parameters PUBLIC "-//OpenKM//DTD Report Parameters 2.1//EN"
                                   "http://www.openkm.com/dtd/report-parameters-2.1.d
<report-parameters>
  <text label="text label" name="consulting_text" width="200px"/>
</report-parameters>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating your own thesaurus

Set the configuration parameters:

Property	Value
kea.thesaurus.owl.file	`vocabulary/inroute.owl`
kea.thesaurus.base.url	`http://www.inroutenetwork.org`
kea.thesaurus.tree.root	`SELECT DISTINCT UID, TEXT FROM {UID} Y {OBJECT}, {UID} rdfs:label {TEXT} ; [rdfs:subClassOf {CLAZZ}] where not bound(CLAZZ) and lang(TEXT)="en" USING NAMESPACE foaf= <http://xmlns.com/foaf/0.1/>, dcterms= <http://purl.org/dc/terms/>, rdf= <http://www.w3.org/1999/02/22-rdf-syntax-ns#>, owl= <http://www.w3.org/2002/07/owl#>, rdfs= <http://www.w3.org/2000/01/rdf-schema#>, skos= <http://www.w3.org/2004/02/skos/core#>, dc= <http://purl.org/dc/elements/1.1/>`
kea.thesaurus.tree.childs	`SELECT DISTINCT UID, TEXT FROM {UID} rdfs:subClassOf {CLAZZ}, {UID} rdfs:label {TEXT} where xsd:string(CLAZZ) = "RDFparentID" and lang(TEXT)="en" USING NAMESPACE foaf= <http://xmlns.com/foaf/0.1/>, dcterms= <http://purl.org/dc/terms/>, rdf= <http://www.w3.org/1999/02/22-rdf-syntax-ns#>, owl= <http://www.w3.org/2002/07/owl#>, rdfs= <http://www.w3.org/2000/01/rdf-schema#>, skos= <http://www.w3.org/2004/02/skos/core#>, dc= <http://purl.org/dc/elements/1.1/>`

Create the inroute.owl file:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
 xmlns:foaf="http://xmlns.com/foaf/0.1/"
 xmlns:dcterms="http://purl.org/dc/terms/"
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:skos="http://www.w3.org/2004/02/skos/core#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">

<rdf:Description rdf:about="http://www.inroutenetwork.org">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Ontology"/>
 <rdfs:comment>INROUTE</rdfs:comment>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Visitors trips and activities</rdfs:label>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Supply and employment</rdfs:label>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_3">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Tourism and territory</rdfs:label>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_4">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Tourism destination management</rdfs:label>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_5">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Research and tools</rdfs:label>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_1">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Visitor</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_2">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Trip</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_3">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Activities</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_4">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Measurement and analysis tools (A.5)</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_5">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Forecast modelling (A.6)</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_6">
```

```
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Other accounting tools (A7)</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_7">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Other terms</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_1_8">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Forms of toruism</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_1"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2_1">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Establishments ( productive )</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_2"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2_2">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Companies</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_2"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2_3">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Tourism industries</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_2"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2_4">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Job</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_2"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.inroutenetwork.org#c_2_5">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
 <rdfs:label xml:lang="en">Other terms</rdfs:label>
 <rdfs:subClassOf rdf:resource="http://www.inroutenetwork.org#c_2"/>
</rdf:Description>

</rdf:RDF>
```

- *Create folder **$TOMCAT_HOME/vocabulary***
- *Copy the file **inroute.owl** into vocabulary folder*
- *Go to Administration -> Utilities -> Thesaurus*
- *On top right click on **Generate button**.*

If you need to convert Skos file to OWL file can use the Skosify Tool.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Database stuff

There are a couple of interesting configuration properties to be set at $TOMCAT_HOME/openkm.properties file.

The parameter hibernate.statistics helps developer enabling the Hibernate statistics. By default this parameter is set to false.

```
hibernate.statistics=true
```

Use these parameters meaningful only for development and fine tuning.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Webservices

OpenKM has API exposed via Webservices with:

- *CMIS*.
- *RESTful*.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CMIS

Content Management Interoperability Services (CMIS) is an open standard that allows different content management systems to inter-operate over the Internet. Specifically, CMIS defines an abstraction layer for controlling diverse document management systems and repositories using web protocols.

OpenKM supports CMIS through the following modules:

- *The Apache Chemistry OpenCMIS library, which is a general-purpose Java library allowing developers to easily write CMIS clients and servers,*
- *Specific OpenKM OpenCMIS connector bundles, allowing OpenKM to be used as a CMIS server.*

There is a CMIS interface available to access the content repository. As defined in the CMIS standard there are several protocols to connect to a CMIS server:

- *Browser: http://localhost:8080/openkm/cmis/browser*
- *AtomPub: http://localhost:8080/openkm/cmis/atom*

For a local OpenKM installation, you can point your browser to http://localhost:8080/openkm/cmis/browser to see the JSON response from the server.+

CMIS clients

- *CMIS Workbench.*
- *http://code.google.com/p/cmisspaces/ is an Adobe AIR client (change the cmisUrl parameter in the file CMISSpacesConfig.xml to point it to OpenKM server).*
- *http://code.google.com/p/cmis-explorer/ is an Adobe AIR client.*

More information at:

- *The official OASIS CMIS TC.*
- *The CMIS 1.0 specification is available in HTML or PDF.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Supported operations

OpenKM connector implements this capabilities from the specification:

Group	*Properties*	
ACL Capabilities	***Name***	***Type***
	ACLs	*MANAGE*
	ACLs propagation	*PROPAGATE*
	Supported permissions	*BASIC*
Filing Capabilities	***Name***	***Supported***
	Multifiling supported	*NO*
	Unfiling supported	*NO*
	Version-specific filing supported	*NO*
Navigation Capabilities	***Name***	***Supported***
	Get descendants supported	*YES*
	Get folder tree supported	*YES*
	Order By supported	*NO*
Object Capabilities	***Name***	***Type***
	Content stream updates	*ANYTIME*
	Changes Object	*NONE*
	Renditions	*NONE*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Type Capabilities	***Name***	***Type***
	Create property types	*NONE*
	New type settable attributes	*controllableACL*
Versioning Capabilities	***Name***	***Type***
	PWC updatable	*NO*
	PWC searchable	*NO*
	All versions searchable	*NO*
Query Capabilities	***Name***	***Type***
	Query	*NONE*
	Joins	*NONE*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create a session

There are three ways to connect to CMIS repository:

Binding Type	Description
Browser	*The fastest and the recommended for **CMIS 1.1** repositories*
Atom	*Recommended for **CMIS 1.0** repositories*
Web services	*The slowest one **and not recommended**.*

Browser binding

Sample to create a connection to the CMIS reposiroty using Browser binding:

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
}
```

The session object created will be used in the following sections to operate with the repository.

Atom binding

Sample to create a connection to the CMIS repository using Atom binding:

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.ATOMPUB_URL, "http://localhost:8080/OpenKM/cmis/atom"
parameters.put(SessionParameter.BINDING_TYPE, BindingType.ATOMPUB.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
}
```

The session object created will be used in the following sections to operate with the repository.

Web services

With this code a user can connect to the repository using web services.

Use Web services is not recommend due slowest connection response.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BINDING_TYPE, BindingType.WEBSERVICES.value());
parameters.put(SessionParameter.WEBSERVICES_ACL_SERVICE, "http://localhost:8080/OpenK
parameters.put(SessionParameter.WEBSERVICES_DISCOVERY_SERVICE, "http://localhost:8080
parameters.put(SessionParameter.WEBSERVICES_MULTIFILING_SERVICE, "http://localhost:80
parameters.put(SessionParameter.WEBSERVICES_NAVIGATION_SERVICE, "http://localhost:808
parameters.put(SessionParameter.WEBSERVICES_OBJECT_SERVICE, "http://localhost:8080/Op
parameters.put(SessionParameter.WEBSERVICES_POLICY_SERVICE, "http://localhost:8080/Op
parameters.put(SessionParameter.WEBSERVICES_RELATIONSHIP_SERVICE, "http://localhost:8
parameters.put(SessionParameter.WEBSERVICES_REPOSITORY_SERVICE, "http://localhost:808
parameters.put(SessionParameter.WEBSERVICES_VERSIONING_SERVICE, "http://localhost:808

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
}
```

The session object created will be used in the following sections to operate with the repository.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Retrieve object samples

This is a list with a bunch of examples about how to retrieve objects in OpenKM using CMIS. For a full specification of CMIS services please visit CMIS documentation

Get root folder

Demonstrate how to get the root folder of the repository.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
    // Get folder and show path
    Folder folder = session.getRootFolder();
    System.out.println(folder.getPath());
}
```

Get object by ID

Every object in the repository have an unique id. This unique id - uuid - is used for retrieving this object.

The "7f5f5726-953b-480a-af9a-24379810dff6" is the unique id - uuid - of the object in the OpenKM repository.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
    // Get object by id
    CmisObject cmisObject = session.getObject("7f5f5726-953b-480a-af9a-24379810dff6")
    System.out.println(cmisObject.getName());
}
```

Get object by path

This code shows how to get an object using the object path.

> *The "**/okm:root/path**" must be a valid OpenKM path and the user have grants for accessing it.*

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
    // Get object by path
    CmisObject cmisObject = session.getObjectByPath("/okm:root/path");
    System.out.println(cmisObject.getName());
}
```

Get folder children

Getting the children of a folder.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();
    Folder folder = session.getRootFolder();
    ItemIterable<CmisObject> children = folder.getChildren();
    for (CmisObject cmisObject: children) {
        System.out.println(cmisObject.getName());
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create object samples

Using CMIS the user can only create Folders and Documents into OpenKM.

Relationships, Policies and Items are not supported in this version.

This is a list with a bunch of examples about how to create objects in OpenKM using CMIS. For a full specification of CMIS services please visit CMIS documentation

*If your repository is in **File Plan Mode** you could create **Record and Email** objects in OpenKM through CMIS services.*

***Record** will be managed using **Folder** object just adding custom properties.*

***Email** will be managed using **Document** object with custom properties.*

Create folder

Sample for creating a folder in OpenKM using CMIS.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();


    // Set properties
    Map<String, Object> properties = new HashMap<String, Object>();
    properties.put(PropertyIds.NAME, "my_folder");
    properties.put(PropertyIds.OBJECT_TYPE_ID, "cmis:folder");

    // Create folder
    Folder newFolder = folder.createFolder(properties);
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create document

Sample for creating a document using CMIS.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();

    // Create a txt file
    String content = "Hello World!";
    String filename = "hello_world.txt";
    String mimetype = "text/plain; charset=UTF-8";
    byte[] contentBytes = content.getBytes("UTF-8");
    ByteArrayInputStream stream = new ByteArrayInputStream(contentBytes);

    // Create content stream
    ContentStream contentStream = session.getObjectFactory().createContentStream(file

    // Create properties object
    Map<String, Object> properties = new HashMap<String, Object>();
    properties.put(PropertyIds.NAME, filename);
    properties.put(PropertyIds.OBJECT_TYPE_ID, "cmis:document");
    // Create document
    Document newDocument = folder.createDocument(properties, contentStream, Versionin
}
```

Copy a document to a folder

Sample for copying an existing OpenKM document into a OpenKM folder.

*The "**3864027b-f181-4291-8e63-d602ee850a8f**" is the unique id - uuid - of the document into OpenKM.*

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();
```

```
// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();

    // Get the document
    CmisObject cmisObject = session.getObject("3864027b-f181-4291-8e63-d602ee850a8f")

    // Copy the document to the folder
    if (cmisObject instanceof Document) {
     Document document = (Document) cmisObject;
     document.copy(folder);
    }
}
```

Update object samples

This is a list with a bunch of examples about how to update objects in OpenKM using CMIS. For a full specification of CMIS services please visit CMIS documentation

Update object properties

This example shows how to change object properties:

The "3864027b-f181-4291-8e63-d602ee850a8f" is the unique id - uuid - of an object in OpenKM repository.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();

    // Get the document
    CmisObject cmisObject = session.getObject("3864027b-f181-4291-8e63-d602ee850a8f")

    // Create properties map
    Map<String, Object> properties = new HashMap<String, Object>();
    properties.put("cmis:name", "new_value.pdf");

    // Update properties
    cmisObject.updateProperties(properties);
}
```

*At the present, the only property that could be changed using CMIS is **cmis:name**. Other properties are not supported for updates.*

Rename object

*If you just want to change the **cmis:name** property, there is a special method.*

The "3864027b-f181-4291-8e63-d602ee850a8f" is the unique id - uuid - of an object in OpenKM repository.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();

    // Get the document
    CmisObject cmisObject = session.getObject("3864027b-f181-4291-8e63-d602ee850a8f")

    cmisObject.rename("new_value.pdf");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Move object samples

This is a list with a bunch of examples about how to move objects in OpenKM using CMIS. For a full specification of CMIS services please visit CMIS documentation

Move an object from a folder to a new folder

The "3864027b-f181-4291-8e63-d602ee850aa3" and "3864027b-f181-4291-8e63-d602ee850a8f" are unique id - uuid - of objects in OpenKM repository.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the folder
    Folder folder = session.getRootFolder();

    Folder final = (Folder) session.getObject("3864027b-f181-4291-8e63-d602ee850aa3")

    // Get the document and move it
    CmisObject cmisObject = session.getObject("3864027b-f181-4291-8e63-d602ee850a8f")
    ((Document) cmisObject).move(folder, final);
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Delete object samples

This is a list with a bunch of examples about how to delete objects in OpenKM using CMIS. For a full specification of CMIS services please visit CMIS documentation

Delete object

The following samples show how to delete an object.

The "3864027b-f181-4291-8e63-d602ee850a8f" is the unique id - uuid - of an object in OpenKM repository.

When the object is a document, all versions of the document are deleted.

When the object is a folder and it is not empty, the folder is not deleted an it raises and exception.

```
// default factory implementation
SessionFactory factory = SessionFactoryImpl.newInstance();
Map<String, String> parameters = new HashMap<String, String>();

// user credentials
parameters.put(SessionParameter.USER, "okmAdmin");
parameters.put(SessionParameter.PASSWORD, "admin");

// connection settings
parameters.put(SessionParameter.BROWSER_URL, "http://localhost:8080/OpenKM/cmis/brows
parameters.put(SessionParameter.BINDING_TYPE, BindingType.BROWSER.value());

// create session
List<Repository> list= factory.getRepositories(parameters);

// List every repository found
for (Repository repository: list) {
    System.out.println("New repository found: " + repository.getId());
    System.out.println("Capabilities: ");
    System.out.println(repository.getCapabilities());
    // Create session
    Session session = repository.createSession();

    // Get the document and move it
    CmisObject cmisObject = session.getObject("3864027b-f181-4291-8e63-d602ee850a8f")
    cmisObject.delete();
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Unsupported and partially supported methods

The list below, describe :

- *Partially supported CMIS methods (they have a not full or distinct behavior rather the CMIS specification).*
- *Unsuported CMIS methods.*

Partially supported

CMIS CmisObject class

- ***updateProperties:*** *This method only allow to change* ***cmis:name*** *property. Other properties will be ignored.*

Unsupported

CMIS Session class

- ***bulkUpdate***

CMIS Document class

- ***addToFolder***
- ***removeFromFolder***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

CmisSync

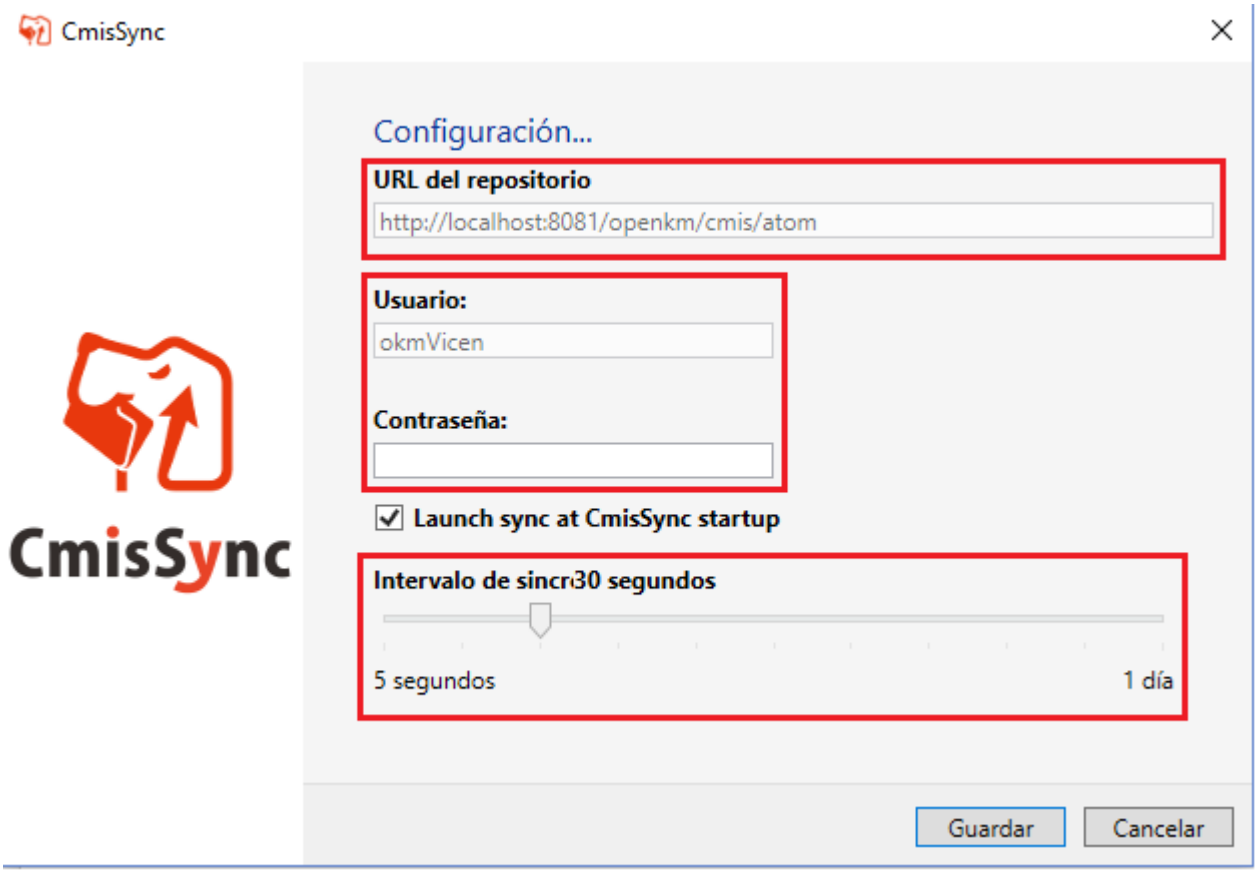
This tool is used to synchronise your documents between the OpenKM server and your local environment.

Requirements

Download CmysSync tool from OpenKM download center

Configuration

During the installation the configuration should be like this:



*Enter the access URL of OpenKM followed by **/cmis/atom**, and enter the username and password with which you login in OpenKM.*

Poll Interval

Recommendations for synchronization interval time configuration:

The default poll interval is 5 seconds. While this is good for tests and environments where you want to get all server changes

immediately, it might prove stressful for the server. You might want to modify the setting to:

- *5 minutes (project collaboration on office files),*
- *6 hours if documents just change a few times per month (sales material, documentation, records management, archives).*

Many people use CmisSync to synchronize from different folders or different servers. In such cases, you might want to configure the poll interval differently for each of them. This is possible: The poll interval is configured at this level.

How to modify the poll interval:

With a text editor like Notepad, open the ***C:\Users\user1\AppData\Roaming\cmissync\config.xml*** *file (replace user1 with your user name. Note: Configure Windows to show hidden files, otherwise you might not see AppData).*

Spot the <folder> tag that defines the synchronized folder you want to tune. Within it, you can see a <pollinterval> tag, probably with the default value of 5000. Change the value to what you want.

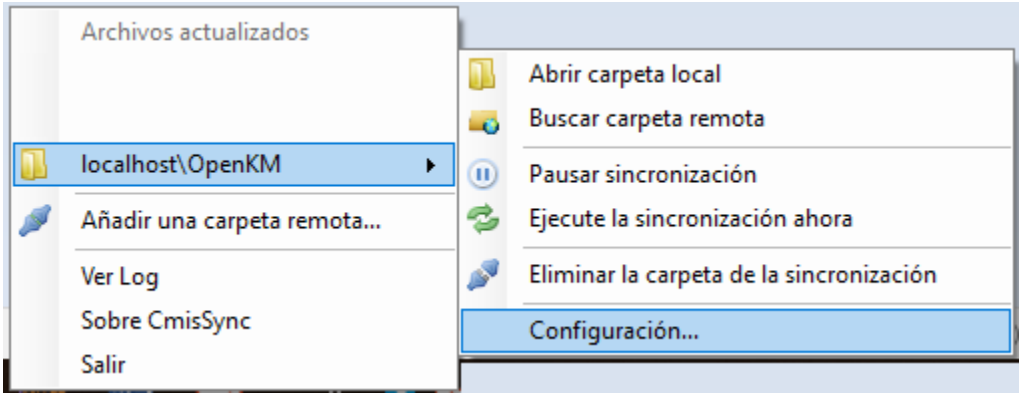
The poll interval is specified in milliseconds, for instance:

- *5 minutes = 300000 milliseconds*
- *6 hours = 21600000 milliseconds*

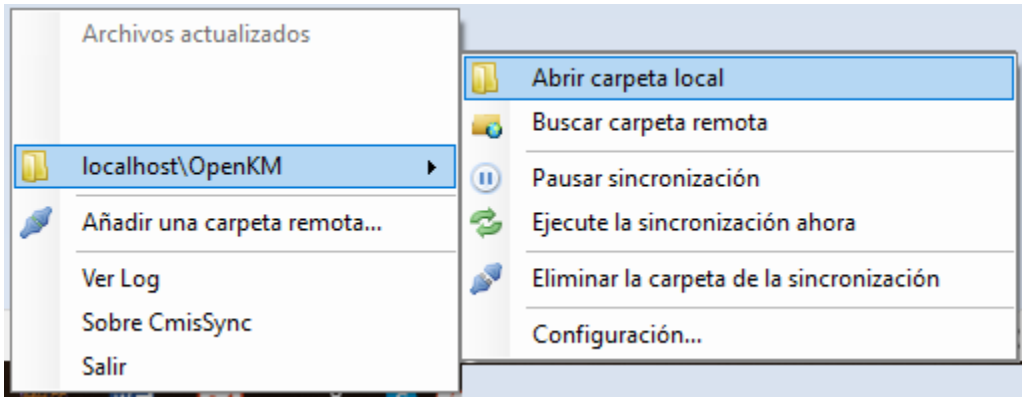
If no <pollinterval> tag exist, create one like this: <pollinterval>300000</pollinterval> just above the next </folder> line.

Contextual Menu

- *You also can access the configuration by right click in the CmisSync icon found in your Windows bottom bar to open the contextual menu:*



- *Also the way for accessing to your local folder:*



Tomcat performance

Compression

Some AtomPub binding calls can generate large XML responses. These responses compress very well. Compressing CMIS AtomPub responses can increase the performance noticeably if the CMIS client supports it. (Apache Chemistry OpenCMIS supports compression since version 0.3.)

Please refer to the documentation of your servlet engine on how to turn compression on.

The following MIME types should be compressed:

application/atomsvc+xml application/atom+xml;type=entry application/atom+xml;type=feed application/cmisquery+xml application/cmisallowableactions+xml application/cmisatom+xml application/cmistree+xml application/cmisacl+xml

To turn on Tomcat compression, edit the file server.xml, which is located under the Tomcat conf directory. The compression configuration is added to the connector element:

```
<Connector port='8080' protocol='HTTP/1.1'
           connectionTimeout='20000'
           redirectPort='8443' URIEncoding='UTF-8'
           compression='on' compressionMinSize='2048'
           compressableMimeType='text/html,text/xml,application/atomsvc+xml,applicati
application/cmisallowableactions+xml,application/cmisatom+xml,application/cmistree+xm
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

RESTful

OpenKM has a complete API exposed via REST. This means you can call any of these API methods from any programming language, like Java, PHP or Python among others. This feature makes it possible to create a custom client, or integrate with third-party applications like a CRM or a CMS.

These URLs are protected by BASIC authentication so you need to provide an user and password to access them.

You can extend the REST API. More information at Creating your own Rest Plugin (extending REST API).

We encourage using our SDK's for accessing OpenKM across REST API rather than you build your own client, more information:

- *The SDK for Java of SDK for JAVA.*
- *The SDK for .NET of SDK for .NET.*

*On almost samples you'll see parameters named "**docId**", "**fldId**" or "**nodeId**". The value of this parameter can be a valid node **UUID (from version 7.x we have deprecated almost methods where path can be used)** .*

Example of nodeId:

- *Using UUID -> "**f123a950-0329-4d62-8328-0ff500fd42db**";*
- *Using path -> "**/okm:root/logo.png**"*

Almost all samples use the command curl, more information at curl tutorial with examples of usage.

Sample usage

*To try these API methods you can use an HTTP Client library or any REST client which ease this process. Or simply you can use the **curl** command-line application. For example, you can list the children folders:*

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
  http://localhost:8080/openkm/rest/folders/de7b07c2-e707-4390-b5cc-cae340c8c3e5/children
```

The result is:

```
[
  {
    "@type": "folder",
    "created": 1568408949000,
    "lastModified": 1568409037000,
    "path": "/okm:root/test/test",
    "author": "okmAdmin",
```

```
    "permissions": 31,
    "uuid": "0fd9f008-27e8-4980-a22b-d71c9a34060a",
    "parent": "de7b07c2-e707-4390-b5cc-cae340c8c3e5",
    "subscribed": false,
    "nodeClass": 0,
    "dispositionCurrentStage": 0,
    "subscriptors": [],
    "keywords": [],
    "categories": [],
    "notes": [],
    "linkTarget": null,
    "actualVersion": {
      "name": "1.1",
      "created": 1568409037000,
      "size": 0,
      "author": "okmAdmin",
      "actual": true,
      "comment": null,
      "checksum": null,
      "action": "PROPERTY_GROUP_ADDED",
      "params": "okg:consulting",
      "uuid": null,
      "parent": "0fd9f008-27e8-4980-a22b-d71c9a34060a"
    },
    "promotedAsRecord": false,
    "hasRelations": false,
    "description": null,
    "restricted": false,
    "nodeClassChildren": [],
    "metadataLabel": null,
    "hasChildren": false,
    "style": 0
  }
]
```

In this case you can see the result in JSON format.

This is a Java client for the same call:

```
package com.openkm;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.Authenticator;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.PasswordAuthentication;
import java.net.URL;

public class JavaRestClient {

    public static void main(String[] args) throws Exception {
        try {
            String fldUuid = "de7b07c2-e707-4390-b5cc-cae340c8c3e5";
            URL url = new URL("http://localhost:8080/openkm/rest/folders/" + fldUuid
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestMethod("GET");
            conn.setRequestProperty("Accept", "application/json");

            Authenticator.setDefault(new Authenticator() {
                protected PasswordAuthentication getPasswordAuthentication() {
                    return new PasswordAuthentication("okmAdmin", "admin".toCharArray
```

```
                }
            });

            if (conn.getResponseCode() == 200) {
                BufferedReader br = new BufferedReader(new InputStreamReader((conn.ge
                System.out.println("Output from Server .... \n");
                String output;

                while ((output = br.readLine()) != null) {
                    System.out.println(output);
                }
            } else {
                System.err.println("Failed : HTTP error code : " + conn.getResponseCo
            }

            conn.disconnect();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Folder samples

Create a new folder:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
 -X POST -H "Content-Type: application/json" -d 'newfolder' \
 http://localhost:8080/openkm/rest/folders/parents/fe239ae2-4d25-4d87-88b2-924d03b87faa
```

Document samples

To create a document, we need to provide the document binary data:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
 -X POST -F content=newDoc.txt -F content=@/home/openkm/doc1.txt \
 http://localhost:8080/openkm/rest/documents/parents/fe239ae2-4d25-4d87-88b2-924d03b87faa
```

Or also from a HTML form:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document Upload</title>
</head>
<body>
    <form method="POST" enctype="multipart/form-data"
          action="http://localhost:8080/openkm/rest/documents/parents/fe239ae2-4d25-4
      Select file: <input type="file" name="content" size="45"/><br/>
      <input type="submit" value="Upload" />
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    </form>
</body>
</html>
```

And now download it:

```
$ curl -u okmAdmin:admin \
  http://localhost:8080/openkm/rest/documents/dab830ea-2fa8-439f-837e-de15409d88b9/content
```

Search samples

Search with parameters:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" -X GET \
  http://localhost:8080/openkm/rest/search?keyword=test&name=doc*&mimeType=application/pdf
```

Search with metadata:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" -X GET \
  http://localhost:8080/openkm/rest/search?keyword=test&property='okp:consulting.name=alfa'
```

Security samples

Show granted users:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" -X GET \
  http://localhost:8080/openkm/rest/auth/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa/permissions/users
```

Show granted roles:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" -X GET \
  http://localhost:8080/openkm/rest/auth/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa/permissions/roles
```

Add a grant:

```
$ curl -v -u okmAdmin:admin -X POST -H "Content-Type: application/x-www-form-urlencoded" \
  -d userId=sochoa -d permissions=15 -d recursive=false \
  http://localhost:8080/openkm/rest/auth/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa/permissions/us
```

Revoke a grant:

```
$ curl -v -u okmAdmin:admin -X DELETE -H "Content-Type: application/x-www-form-urlencoded" \
  -d permissions=15 -d recursive=false \
  http://localhost:8080/openkm/rest/auth/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa/permissions/users/sochoa
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Metadata samples

The metadata group definition used in the samples:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"
                                 "http://www.openkm.com/dtd/property-groups-2.0.dtd">
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Name" type="text" name="okp:consulting.name" >
      <validator type="req"/>
    </input>
    <textarea label="Comment" name="okp:consulting.comment" >
      <validator type="req"/>
    </textarea>
  </property-group>
</property-groups>
```

Add a metadata group:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
 -X POST -H "Content-type: application/json"
 -d '{"properties":[{"name":"okp:consulting.comment","value":"new comment"},
{"name":"okp:consulting.name","value":"new name"},
{"name":"okp:consulting.date","value":"20190917163517"}],"grpName":"okg:consulting"}' \
 http://localhost:8080/openkm/rest/propertygroups/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa/groups
```

Set metadata group values:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
 -X PUT -H "Content-Type: application/json" \
 -d '[{"name":"okp:consulting.comment","value":"set comment"},{"name":"okp:consulting.name","value":"set name"},
{"name":"okp:consulting.date","value":"20190917164717"}]' \
 http://localhost:8080/openkm/rest/propertygroups/nodes/fe239ae2-4d25-4d87-88b2-
924d03b87faa/groups/okg:consulting/properties
```

Notes samples

Create a note:

```
$ curl -u okmAdmin:admin -H "Accept: application/json" \
 -X POST -H "Content-Type: application/x-www-form-urlencoded" -d 'text=Hello, world!' \
 http://localhost:8080/openkm/rest/note/nodes/fe239ae2-4d25-4d87-88b2-924d03b87faa
```

Conversion samples

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

With conversion feature from the OpenKM API, you can convert, for example, a document from MS Office Word format to PDF, among other conversion types available.

Convert a doc file to pdf:

```
$ curl -u okmAdmin:admin -F content=@sample.doc -o sample.pdf \
  http://localhost:8080/openkm/rest/conversion/doc/pdf
```

Or also from a HTML form:

```
<html>
  <body>
    <form method="POST" enctype="multipart/form-data"
          action="http://localhost:8080/OpenKM/services/rest/conversion/doc2pdf">
      Select file: <input type="file" name="content" size="45"/><br/>
      <input type="submit" value="Convert" />
    </form>
  </body>
</html>
```

Convert a doc file to txt:

```
$ curl -u okmAdmin:admin -F content=@sample.doc http://localhost:8080/openkm/rest/conversion/doc/txt
```

Convert an image file to txt:

```
$ curl -u okmAdmin:admin -F content=@sample.tif http://localhost:8080/rest/rest/conversion/img/txt
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Swagger

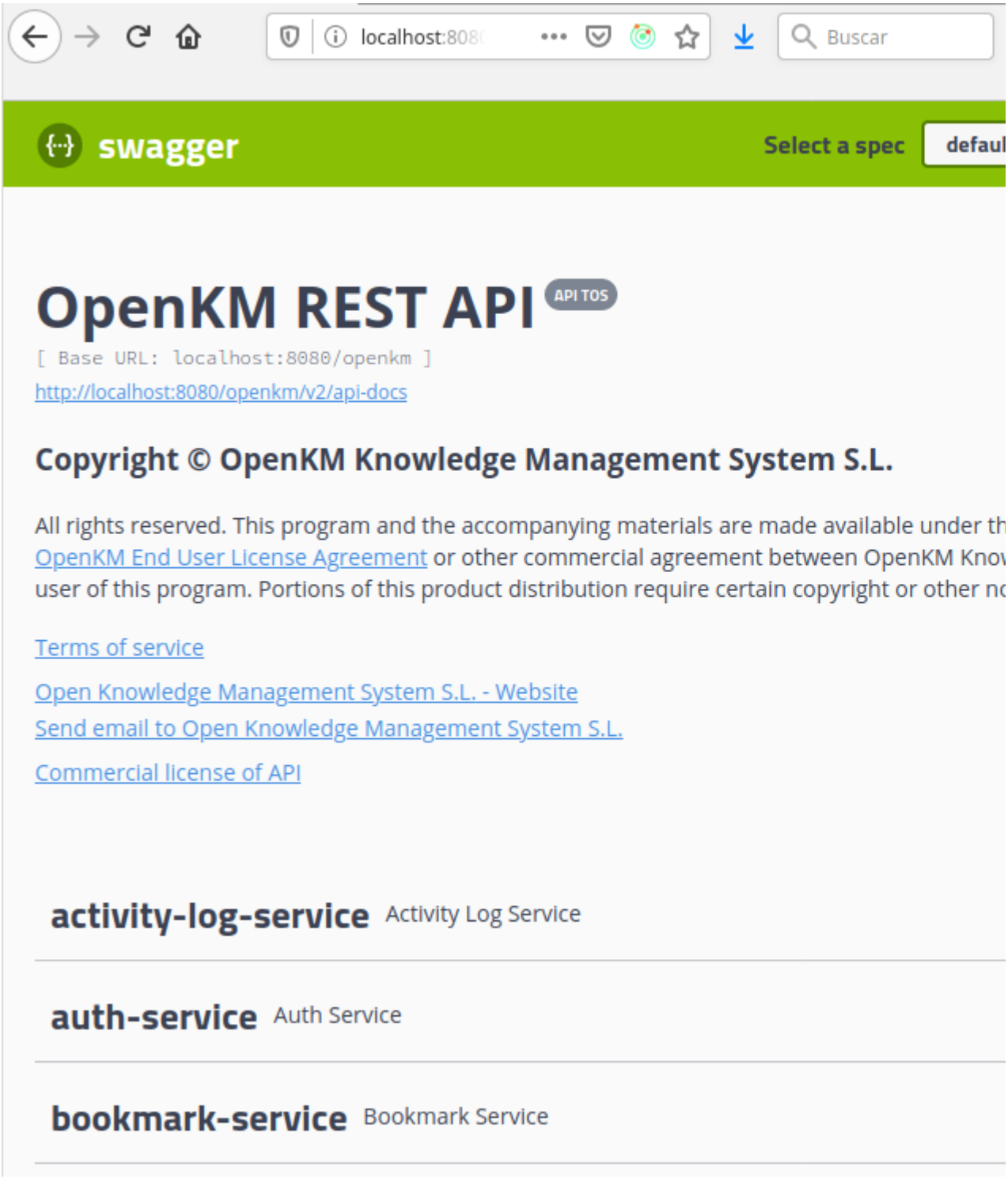
***Swagger** is an open-source software framework backed by a large ecosystem of tools that helps developers design, build, document, and consume RESTful Web services. While most users identify Swagger by the Swagger UI tool, the Swagger toolset includes support for automated documentation, code generation, and test case generation.*

How to accessing Swagger

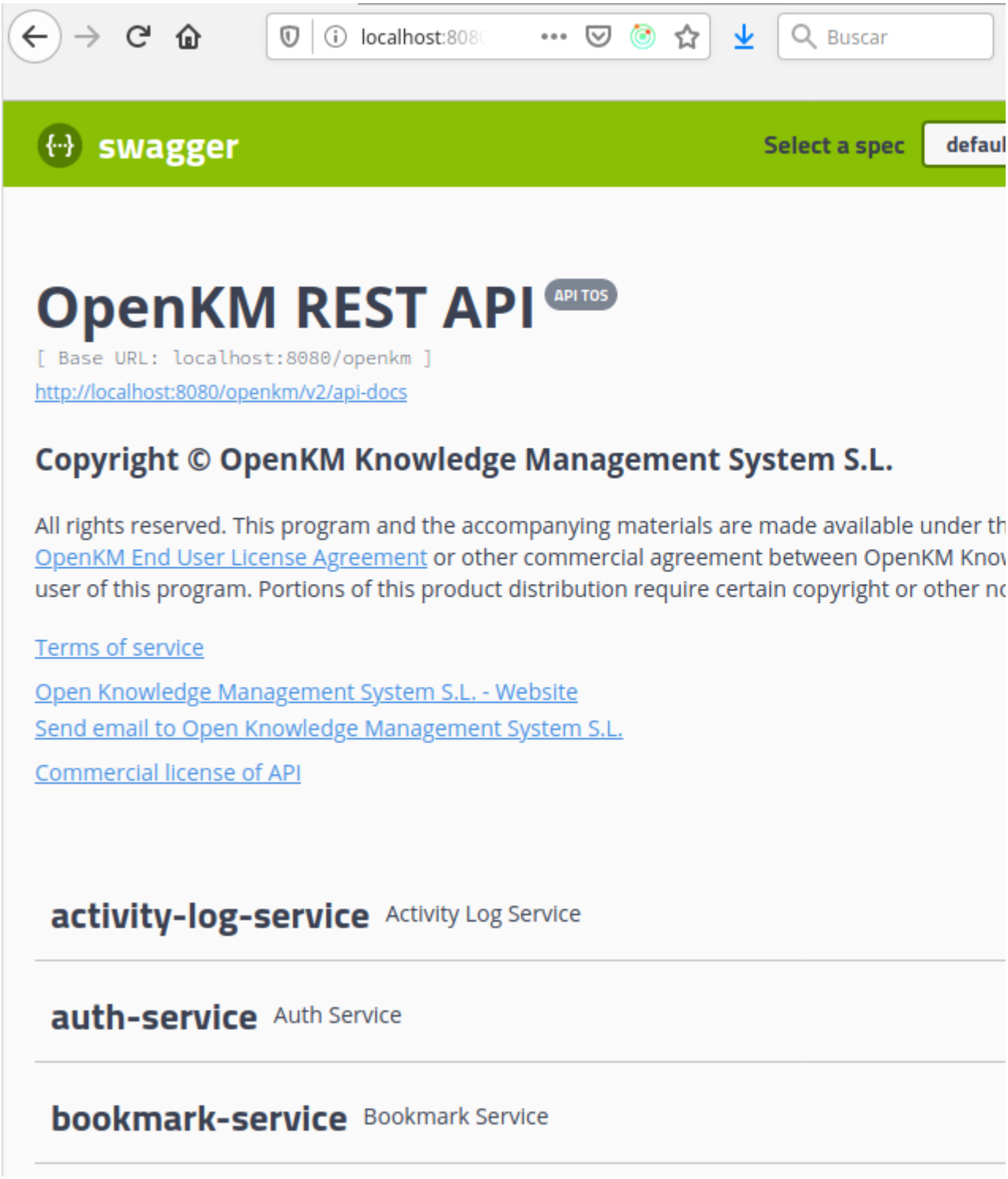
*In the **sample below we use our OpenKM demo URL**. You **should use your own OpenKM URL**.*

For accessing Swagger documentation, click in the URL http://localhost:8080/openkm/swagger-ui.html it will be shown and screen like shown below asking for authorized user credentials:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Then will be shown the Swagger tool:

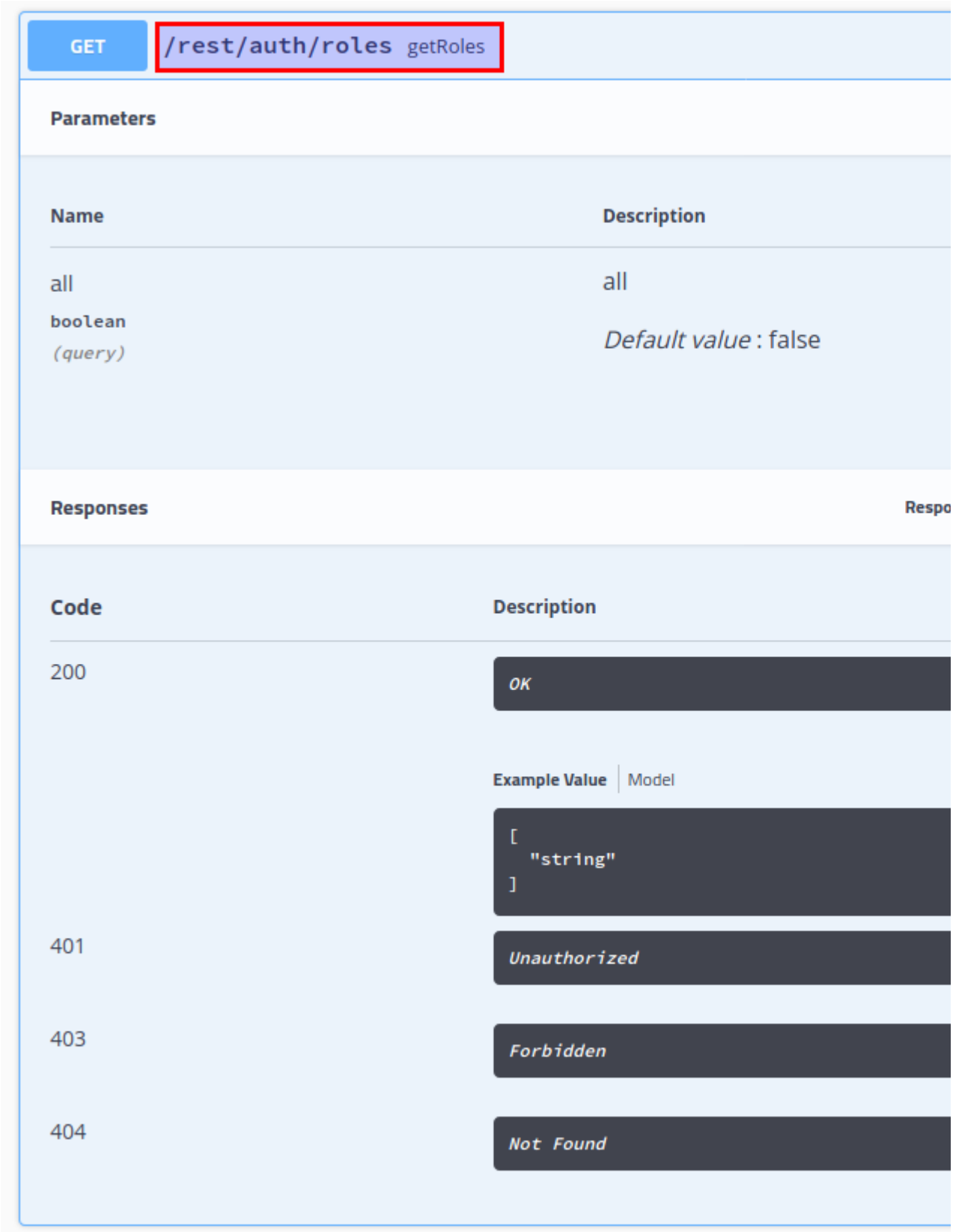


Playing with Swagger

*Let's do a **quick test** of the tool **checking the getRoles** method **from auth-service**.*

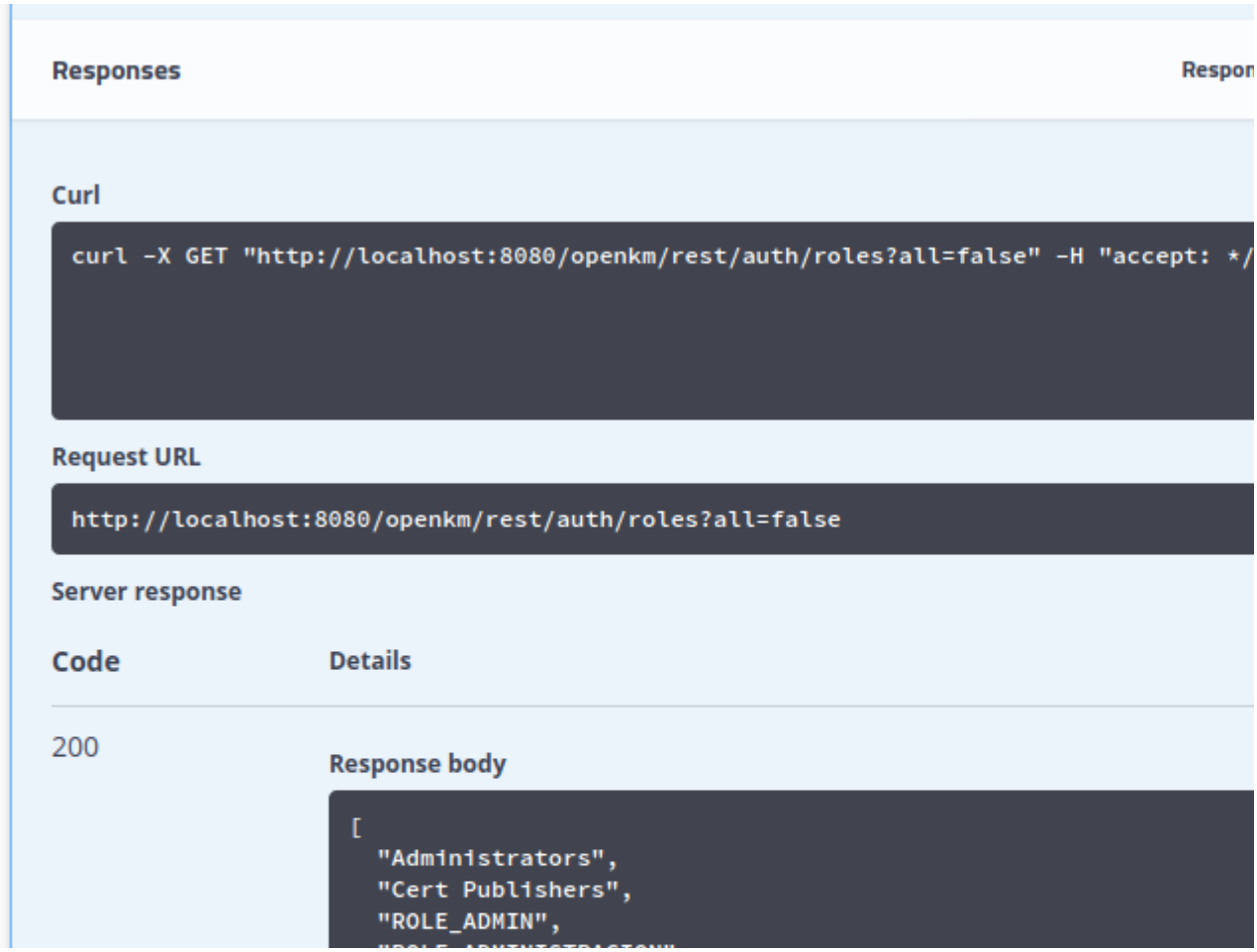
- ***Open the auth-service***

- ***Open the "roles"** method and **click on the button "Try it out"**:*



- ***Click on the "Execute" button;***

- *Finally will be shown the results*



```
  "ROLE_ADMINISTRACION",
  "ROLE_LICENSE",
  "ROLE_MARKETING",
  "ROLE_PROYECTO",
  "ROLE_SECURITY_RECURSIVE",
  "ROLE_TASK_MANAGER_ADMIN",
  "ROLE_TECNICO",
  "ROLE_USER"
]
```


Response headers


```
cache-control: no-cache, no-store, max-age=0, must-revalidate
content-type: application/json;charset=UTF-8
date: Wed, 12 Feb 2020 13:04:33 GMT
expires: 0
pragma: no-cache
transfer-encoding: chunked
x-application-context: application:none
x-content-type-options: nosniff
x-frame-options: DENY
x-xss-protection: 1; mode=block
```


Responses

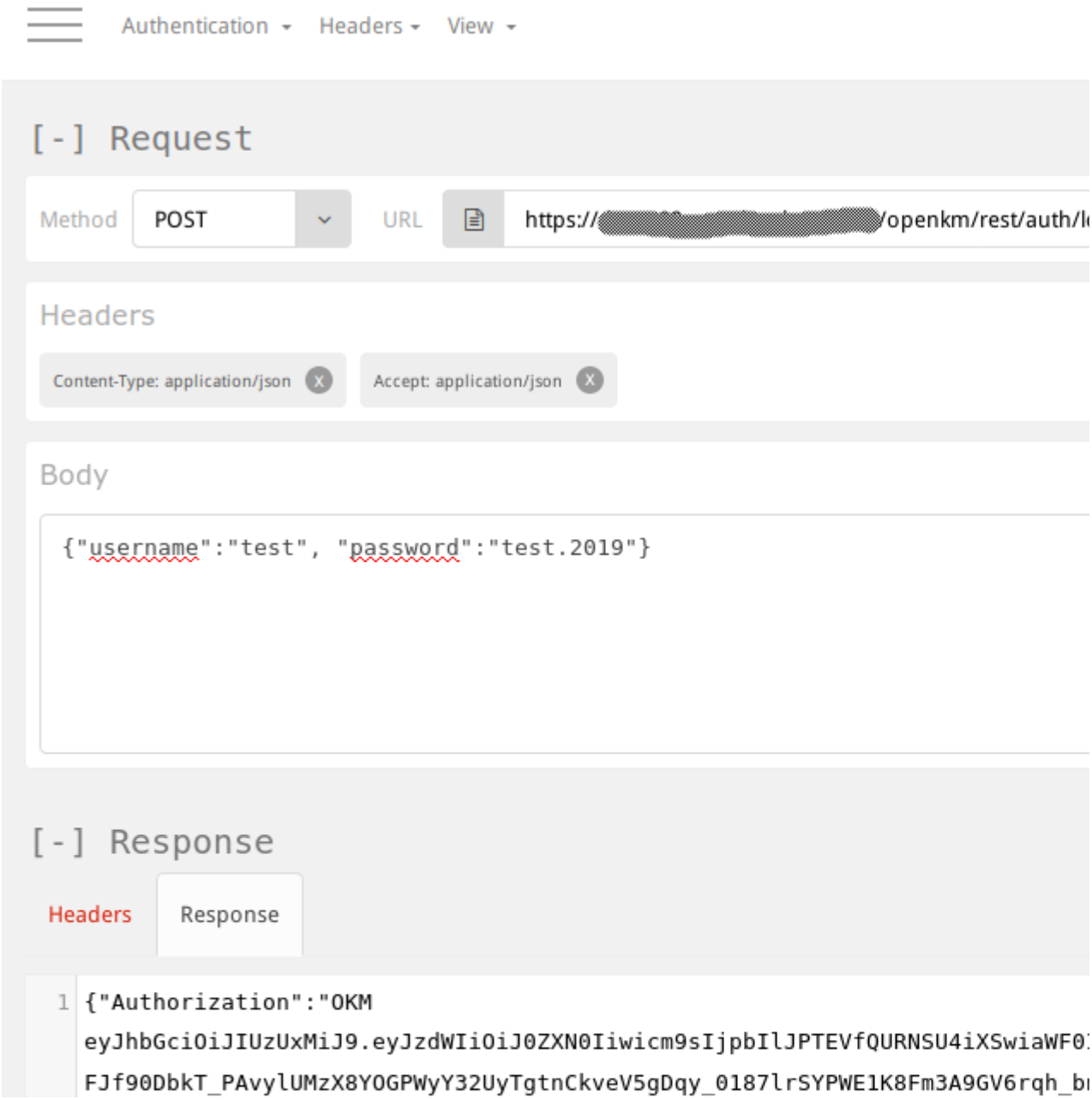
Code	Description
200	*OK* **Example Value** | Model `[ "string" ]`
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

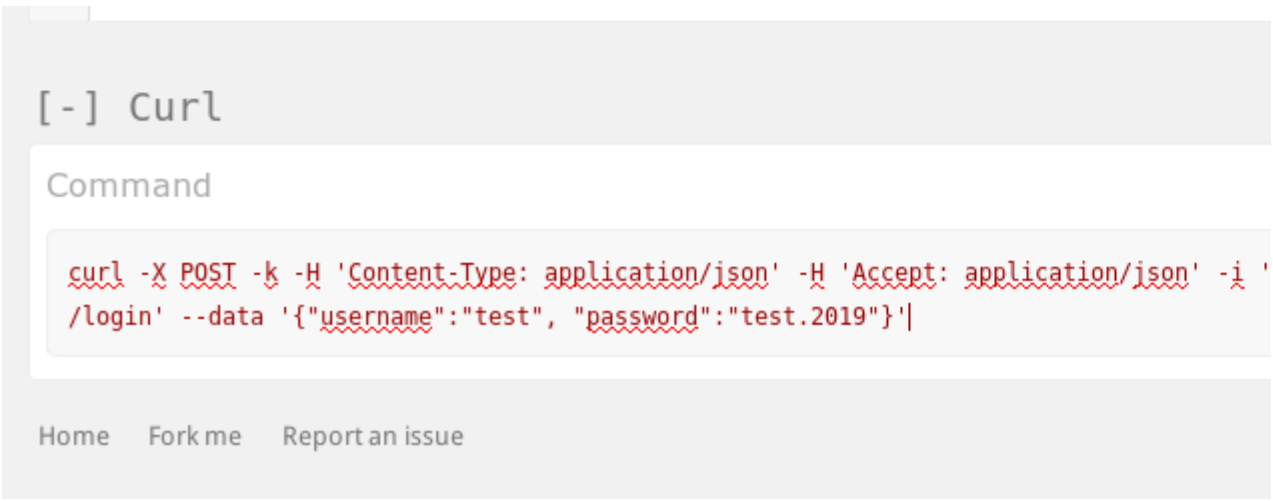
RestClient sample

RESTClient addon for mozilla is a debugger for RESTful web services.

Login

- *First might login to get the authorization token what might be used in the other calls.*
- *The **username** and **password** is **sent in the body**.*
- *The **content-type** is set to "**application**/json" in the header.*
- *The **accept** is set to "**application**/json" in the header.*

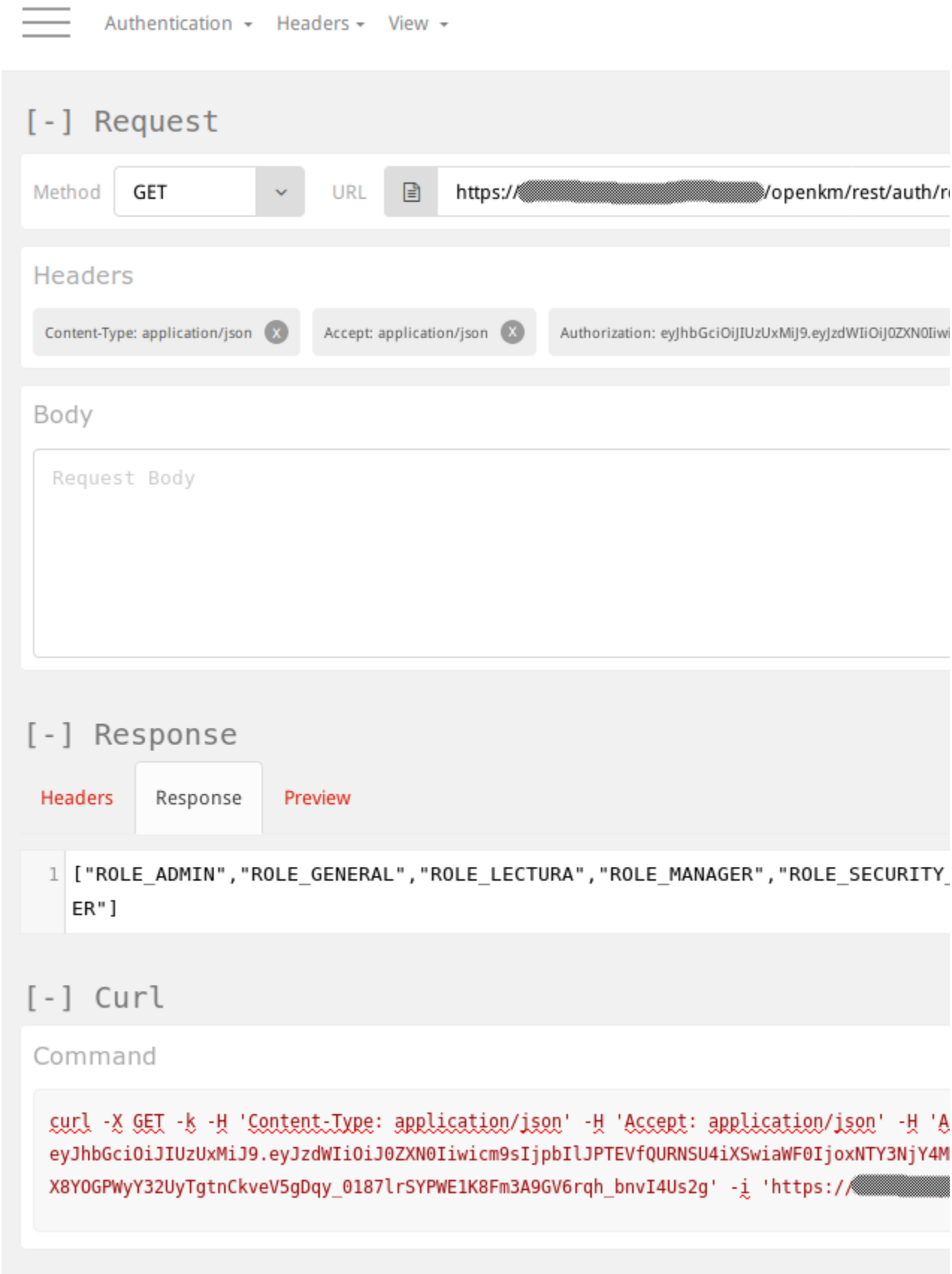




Get roles

- *The **authorization** token is set in the header.*
- *The **content-type** is set to "**application**/json" in the header.*
- *The **accept** is set to "**application**/json" in the header.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Extending frontend UI

To extend frontend you need to get the source code and compile the changes into.

The OpenKM plugin extensions makes it easy to extend the OpenKM end user interface by encapsulating plugin code and making it reusable between OpenKM versions.

The OpenKM plugin extension architecture is based on:

- *Extensions*
- *Events*
- *Handlers*
- *Communicators*

***Extensions are available widget definitions** that allows the developer to make extensible panels and widgets (for example adding new tab panel on tab document)*

***Events are a collection of events that OpenKM UI** triggers each time any changes occur (for example when a new keyword is added in the tab document the triggered event HasDocumentEvent.KEYWORD_ADDED)*

***Handlers are a collection of methods called internally by OpenKM** . Handlers must be implemented into your extensions in order to collect OpenKM triggered events. Each extension you make can have one or several handlers, that are automatically registered by OpenKM on the loading process. OpenKM internally triggers events to each declared handler.*

***Comunicators are a collection of methods available as OpenKM Comunitators API** to access transparently some internal UI values. There are several Communicators, for example with GeneralComunicator you can access some general actions like refreshing UI as GeneralComunicator.refreshUI(). OpenKM Communicators API has read and write methods to interact with internal OpenKM UI objects.*

Creating OpenKM plugin extensions is easy but you need some java knowledge and pay special attention to the Google Web ToolKit API that's used to build OpenKM UI. http://code.google.com/webtoolkit/

Enable sample extensions

*OpenKM comes with some frontend UI sample extensions not enabled by default. To enable them go to **Administration > Utilies** > Database query and execute the SQL queries:*

Extension	*SQL sentence*
ToolBarButtonExample	`INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('9f84b330-d`

TabWorkspaceExample	`INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('44f94470-d`
TabFolderExample	`INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('d95e01a0-d`
TabFolderExample	`INSERT INTO OKM_EXTENSION (EXT_UUID,EXT_NAME) VALUES('d9dab640-d`

After registering a sample extension, you also need to enable the from ***Administration*** > ***Profiles***.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Frontend UI hello world sample

Description

This example add a new Widget in tab documents user interface:

- *Create a new GWT module.*
- *Import new GWT module into default OpenKM frontend GWT module.*
- *Create a new basic widget.*
- *Use widget into OpenKM frontend UI.*

Sample

Create a new GWT module

Create a file called ***HelloWorld.gwt.xml*** *into* ***src/main/java/com/openkm/extension/frontend***

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE module SYSTEM "http://google-web-toolkit.googlecode.com/svn/releases/2.0/di
<module>
  <!-- Inherit the core Web Toolkit stuff -->
  <inherits name='com.google.gwt.user.User'/>
  <inherits name="com.google.gwt.http.HTTP"/>
</module>
```

Add module into OpenKM frontend UI

Edit ***Customization.gwt.xml*** *into* ***src/main/java/com/openkm/extension/frontend*** *and add the new module*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE module SYSTEM "http://google-web-toolkit.googlecode.com/svn/releases/2.0/di
<module>
  <!-- Inherit the extension widgets -->
  <inherits name="com.openkm.extension.frontend.HelloWorld"/>
</module>
```

Create new widget

Create a file called ***HelloWorld.java*** *into* ***src/main/java/com/openkm/extension/frontend/client***

```
package com.openkm.extension.frontend.client;

import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.Button;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.extension.communicator.GeneralCommunicator;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.frontend.client.extension.communicator.TabDocumentCommunicator;
import com.openkm.frontend.client.extension.event.HasDocumentEvent;
import com.openkm.frontend.client.extension.event.HasDocumentEvent.DocumentEventConst
import com.openkm.frontend.client.extension.event.HasLanguageEvent;
import com.openkm.frontend.client.extension.event.HasLanguageEvent.LanguageEventConst
import com.openkm.frontend.client.extension.event.handler.DocumentHandlerExtension;
import com.openkm.frontend.client.extension.event.handler.LanguageHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabdocument.TabDocumentExtension;

public class HelloWorld extends TabDocumentExtension implements DocumentHandlerExtens

    Button refresh;
    VerticalPanel vPanel;

    public HelloWorld() {
        HTML html = new HTML("Hello Word");
        refresh = new Button("refresh UI");
        refresh.addClickHandler(new ClickHandler() {
            @Override
            public void onClick(ClickEvent event) {
                GeneralCommunicator.refreshUI();
            }
        });

        vPanel = new VerticalPanel();
        vPanel.add(html);
        vPanel.add(refresh);

        refresh.setStyleName("okm-Input");

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "Hello tab";
    }

    @Override
    public void onChange(DocumentEventConstant event) {
        if (event.equals(HasDocumentEvent.DOCUMENT_CHANGED)) {
            Window.alert("document changed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.KEYWORD_ADDED)) {
            Window.alert("keyword added - " + event.getType());
        } else if (event.equals(HasDocumentEvent.KEYWORD_REMOVED)) {
            Window.alert("keyword removed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.CATEGORY_ADDED)) {
            Window.alert("category added - " + event.getType());
        } else if (event.equals(HasDocumentEvent.CATEGORY_REMOVED)) {
            Window.alert("category removed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.TAB_CHANGED)) {
            Window.alert("tab changed - " + event.getType() + " - actual tab " + TabD
        } else if (event.equals(HasDocumentEvent.PANEL_RESIZED)) {
            Window.alert("panel resized - " + event.getType());
        } else if (event.equals(HasDocumentEvent.SECURITY_CHANGED)) {
            Window.alert("security changed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.NOTE_ADDED)) {
            Window.alert("note added - " + event.getType());
        }
    }

    @Override
    public void onChange(LanguageEventConstant event) {
        if (event.equals(HasLanguageEvent.LANGUAGE_CHANGED)) {
            Window.alert("language changed");
        }
    }
```

```
}
```

Use widget into OpenKM

*Edit **Customization.java** OpenKM class and add the HelloWorld widget:*

```
package com.openkm.extension.frontend.client;

import java.util.ArrayList;
import java.util.List;

public class Customization {

    /**
     * getExtensionWidgets
     */
    public static List<Object> getExtensionWidgets(List<String> uuidList) {
        List<Object> extensions = new ArrayList<>();

        // add here your widget extensions
        extensions.add(new HelloWorld());

        return extensions;
    }

}
```

Results

Compile project, login into OpenKM and you'll see something like this:

Frontend UI extensions

Extensions are available widget definitions *that allows the developer to make extensible panels and widgets (for example adding new tab panel on tab document).*

Available extensions sections are:

Widget	*Workspace*	*Description*
Workspace	***Desktop***	***Location:*** *Top right.* *Usually they are desktop, search, dashboard and administrator tabs.* *More information at TabWorkspaceExtension.*
Tab document.	***Desktop***	***Location:*** *Bottom right.* *Usually they are properties, notes, security, history, and preview tabs.* *More information at TabDocumentExtension.*
Tab folder	***Desktop***	***Location:*** *Bottom right.* *Usually they are properties, notes and security tabs.* *More information at TabFolderExtension.*
Tab mail	***Desktop***	***Location:*** *Bottom right.* *Usually they are properties, notes, security and preview tabs.* *More information at TabMailExtension.*
Tab record	***Desktop***	***Location:*** *Bottom right.* *Usually they are properties, notes and security tabs.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

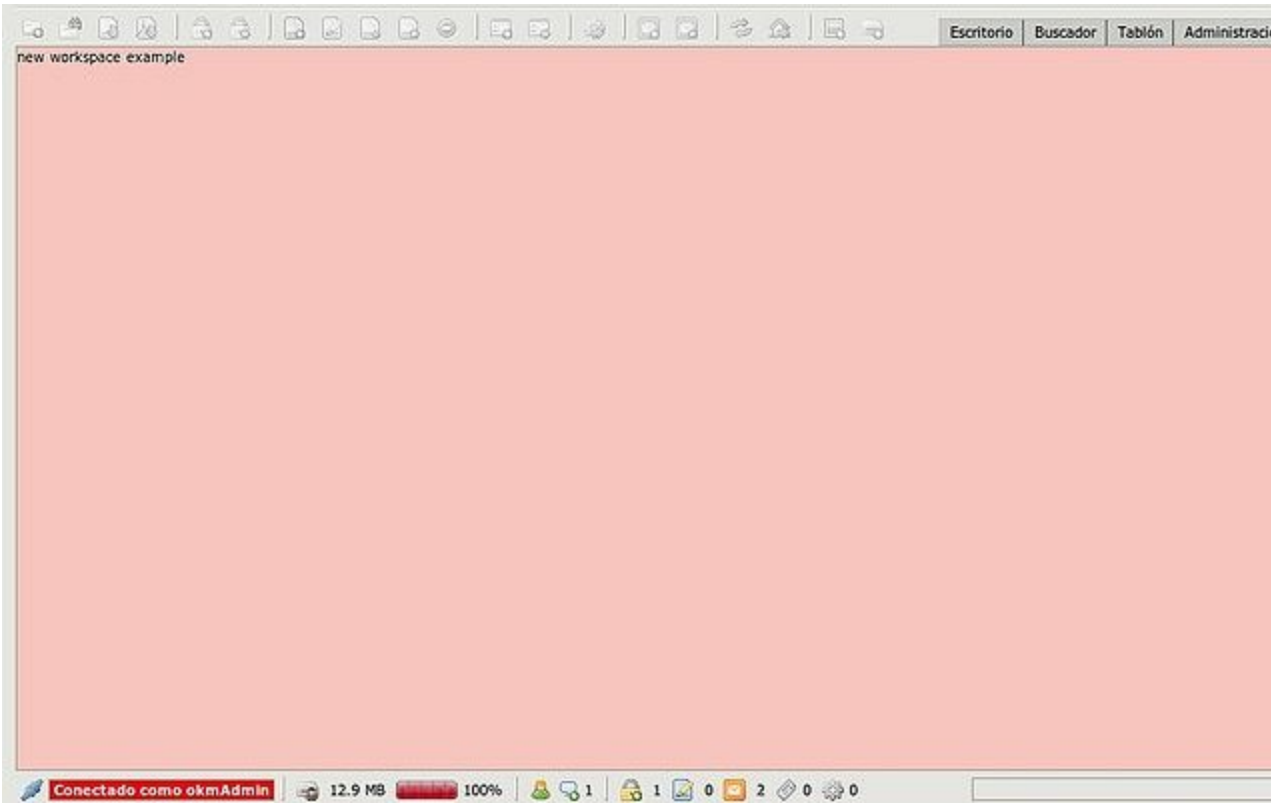
		More information at TabRecordExtension.
Toolbar box	***Dashboard***	***Location:*** *Dashboard toolbar.* *Toolbar icons* *More information at ToolBarBoxExtension.*
Toolbar button	***Desktop***	***Location:*** *Top center.* *Toolbar icons of the OpenKM features like create folder, delete, upload new file, etc.* *More information at ToolBarButtonExtension.*
Menu Bar	***Desktop***	***Location:*** *Top.* *Usually the menu bar options are file, edit, tools, bookmarks, templates and help menus.* *More information at MenuBarExtension.*
Menu Item	***Desktop***	***Location:*** *Top.* *Here are set the menu options of each main menu bar.* *More information at MenuItemExtension.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Widget library

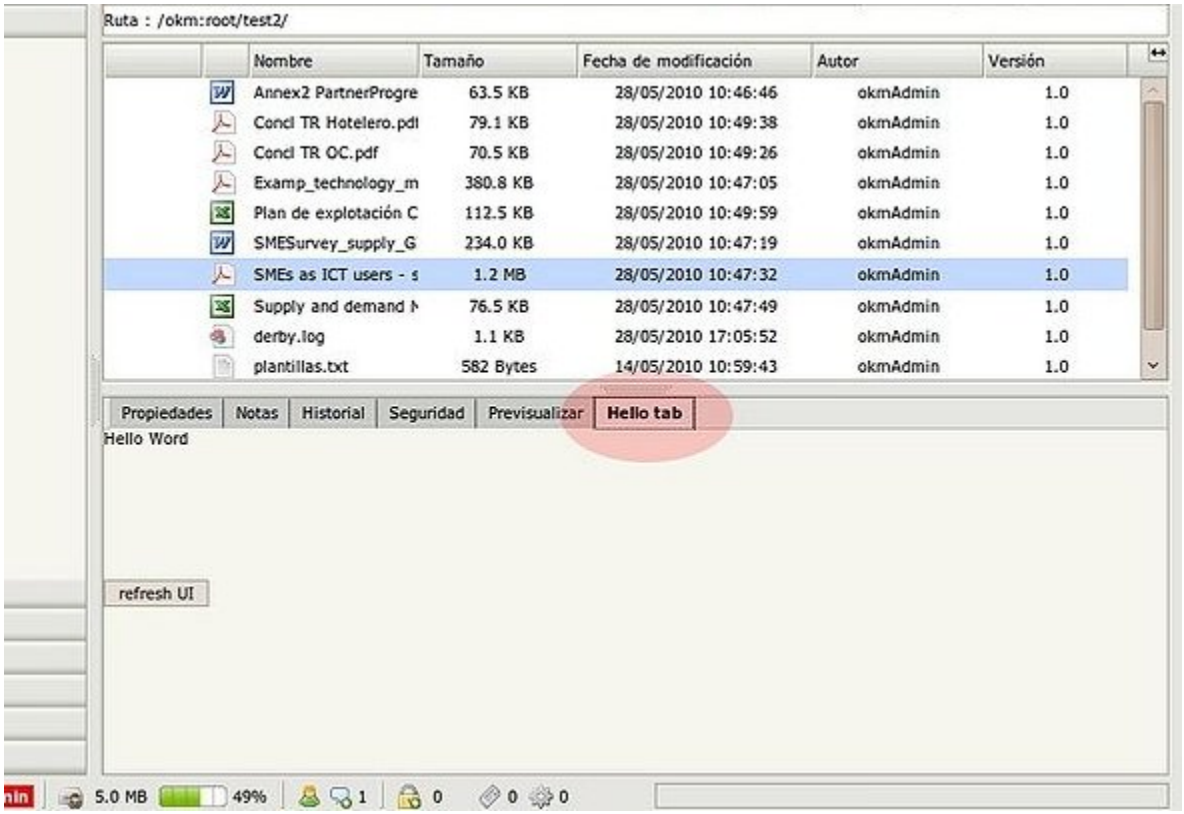
Tab workspace

> *More information at: TabWorkspaceExtension.*



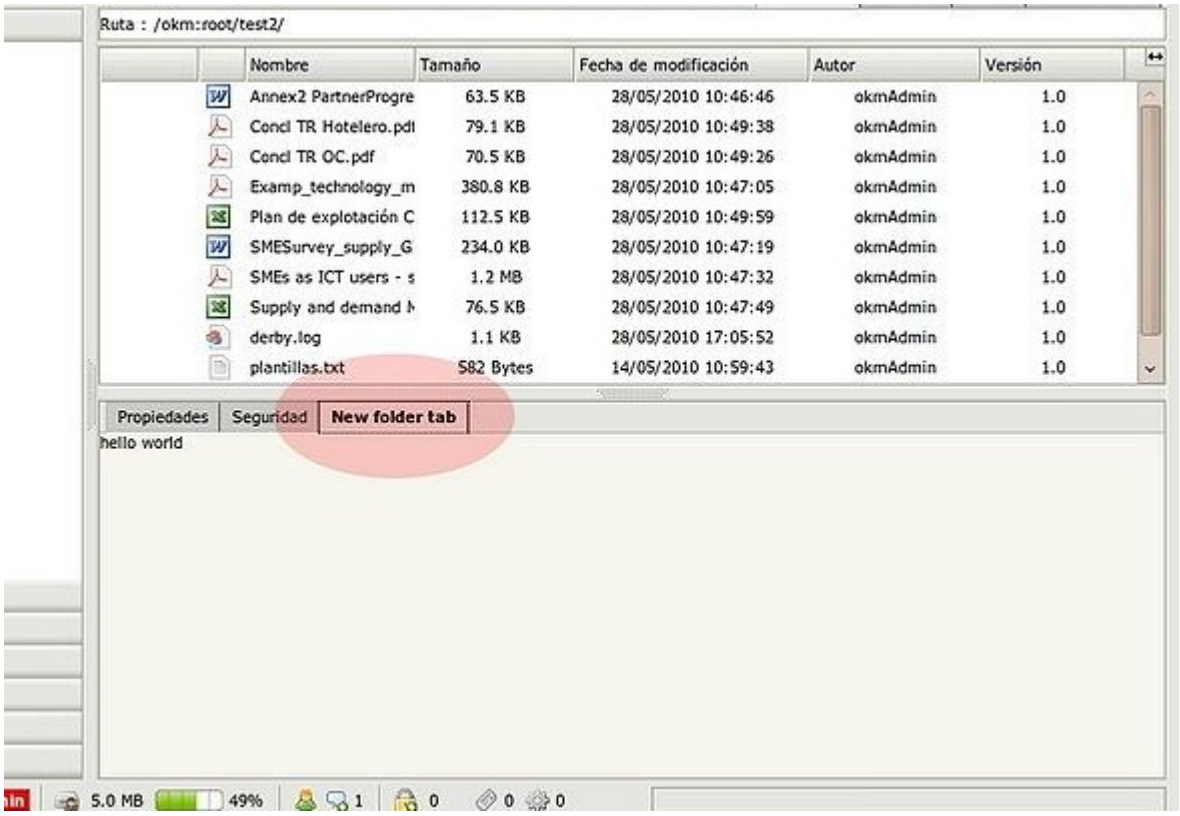
Tab Document

> *More information at: TabDocumentExtension.*



Tab folder

More information at: TabFolderExtension.



Main menu

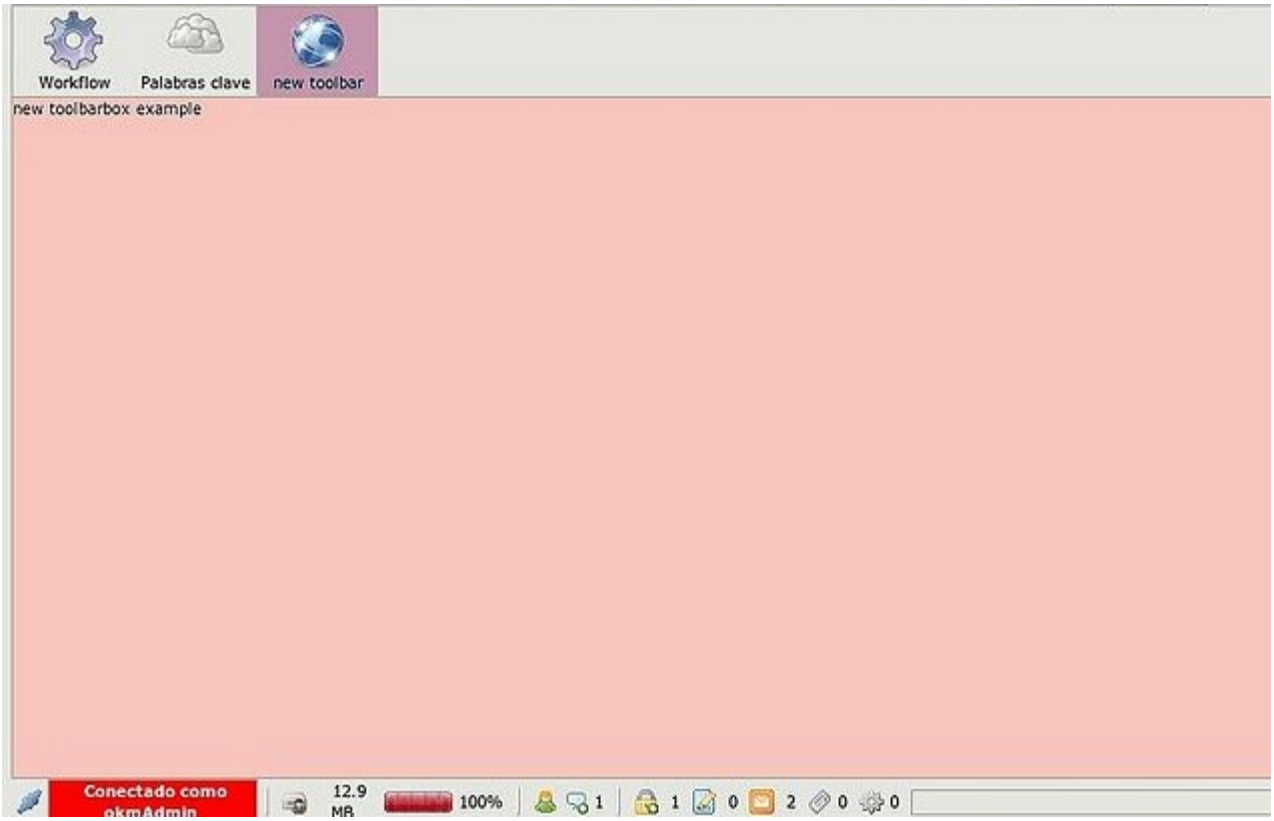
> *More information at: MenuBarExtension and MenuItemExtension.*

Toolbar button

> *More information at: ToolBarButtonExtension.*

Dashboard toolbar button

More information at: ToolBarBoxExtension.



TabWorkspace Extension

Adds a new tab at top right workspace section (usually there are shown desktop, search, dashboard and administration tab).

Methods

method	Description
getTabText()	*Used by OpenKM to get the tab text.*
setTab(TabBar tabBar, int tabIndex)	*Used by openkm to set tabBar and tab index.*

Basic sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.TabBar;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.extension.widget.tabworkspace.TabWorkspaceExtension

public class TabWorkspaceExample extends TabWorkspaceExtension {

    private VerticalPanel vPanel;

    /**
     * TabWorkspaceExample
     */
    public TabWorkspaceExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("new workspace example"));
        vPanel.setStyleName("okm-Input");

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "tab workspace";
    }

    @Override
    public void setTab(TabBar tabBar, int tabIndex) {
    }
}
```

Sample with iframe

```
package com.openkm.extension.frontend.client;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.google.gwt.user.client.DOM;
import com.google.gwt.user.client.ui.Frame;
import com.google.gwt.user.client.ui.HasAlignment;
import com.google.gwt.user.client.ui.TabBar;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.Main;
import com.openkm.frontend.client.extension.widget.tabworkspace.TabWorkspaceExtension

public class TabWorkspace extends TabWorkspaceExtension {

    private VerticalPanel vPanel;
    private Frame iframe;
    private String textLabel = "";
    private TabBar tabBar;
    private int tabIndex = 0;

    /**
     * TabWorkspace
     */
    public TabWorkspace() {
        vPanel = new VerticalPanel();
        iframe = new Frame("about:blank");

        DOM.setElementProperty(iframe.getElement(), "frameborder", "0");
        DOM.setElementProperty(iframe.getElement(), "marginwidth", "0");
        DOM.setElementProperty(iframe.getElement(), "marginheight", "0");

        // Commented because on IE show clear if allowtransparency=true
        DOM.setElementProperty(iframe.getElement(), "allowtransparency", "false");
        DOM.setElementProperty(iframe.getElement(), "scrolling", "no");

        iframe.setUrl(Main.CONTEXT + "/sample/index.jsp");
        iframe.setStyleName("okm-Iframe");

        vPanel.add(iframe);
        vPanel.setCellHorizontalAlignment(iframe, HasAlignment.ALIGN_CENTER);

        vPanel.setWidth("100%");
        vPanel.setHeight("100%");

        initWidget(vPanel);
    }

    /**
     * Sets the size on initialization
     *
     * @param width The max width of the widget
     * @param height The max height of the widget
     */
    public void setPixelSize(int width, int height) {
        iframe.setPixelSize(width - 2, height - 2);
    }

    /**
     * setTextLabel
     */
    public void setTextLabel(String textLabel) {
        this.textLabel = textLabel;
    }

    @Override
    public String getTabText() {
        return textLabel;
    }

    @Override
    public void setTab(TabBar tabBar, int tabIndex) {
```

```
            this.tabBar = tabBar;
            this.tabIndex = tabIndex;
        }

    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

WorkspaceHandlerExtension

Any extension that implements WorkspaceHandlerExtension will be looking at all the events fired from the workspace tab panel.

Methods

method	Description
onChange(WorkspaceEventConstant event)	*Each time it'll be some new tab panel event the method onChange it'll be triggered by OpenKM.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.TabBar;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.extension.event.HasWorkspaceEvent.WorkspaceEventCon
import com.openkm.frontend.client.extension.event.handler.WorkspaceHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabworkspace.TabWorkspaceExtension

public class TabWorkspaceExample extends TabWorkspaceExtension implements WorkspaceHa

    private VerticalPanel vPanel;

    /**
     * TabWorkspaceExample
     */
    public TabWorkspaceExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("new workspace example"));
        vPanel.setStyleName("okm-Input");

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "tab workspace";
    }

    @Override
    public void setTab(TabBar tabBar, int tabIndex) {
    }

    @Override
    public void onChange(WorkspaceEventConstant event) {
        Window.alert("workspace event fired");
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

WorkspaceComunicator

Use it to retrieve workspace data.

Methods

method	return	type	Description
getSelectedTab()	***int***	***read***	*Get the selected tab id.*
changeSelectedTab(int selectedTab)	***void***	***write***	*Change the selected tab.*
getSelectedWorkspace()	***int***	***read***	*Get selected workspace id.*
getWorkspace()	***GWTWorkspace***	***read***	*Get selected workspace object.*
getTabExtensionIndex(TabWorkspaceExtension widget)	***int***	***read***	*Get the tab id of the widget.*

Sample

```
int tab = WorkspaceComunicator.getSelectedTab();
if (tab == 0) {
  Window.alert("First tab enabled");
} else {
  Window.alert("Other stack");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabDocument Extension

Add a new tab at bottom right (usually there are shown properties, notes, security, history, and preview tabs) in the document tab section.

Methods

method	Description
getTabText()	*Used by OpenKM to get the tab text.*

Basic sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.extension.widget.tabdocument.TabDocumentExtension;

public class TabDocumentExample extends TabDocumentExtension {

    VerticalPanel vPanel;

    public TabDocumentExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("Content tab example"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New document tab";
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

DocumentHandlerExtension

Any extension that implements DocumentHandlerExtension will look at all the events triggered from the document tab.

Methods

method	Description
onChange(DocumentEventConstant event)	*Each time there is a new document event, the method onChange will be triggered by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.extension.comunicator.TabDocumentComunicator;
import com.openkm.frontend.client.extension.event.HasDocumentEvent;
import com.openkm.frontend.client.extension.event.HasDocumentEvent.DocumentEventConst
import com.openkm.frontend.client.extension.event.handler.DocumentHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabdocument.TabDocumentExtension;

public class TabDocumentExample extends TabDocumentExtension implements DocumentHandl

    VerticalPanel vPanel;

    public TabDocumentExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("Content tab example"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New document tab";
    }

    @Override
    public void onChange(DocumentEventConstant event) {
        if (event.equals(HasDocumentEvent.DOCUMENT_CHANGED)) {
            Window.alert("document changed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.KEYWORD_ADDED)) {
            Window.alert("keyword added - " + event.getType());
        } else if (event.equals(HasDocumentEvent.KEYWORD_REMOVED)) {
            Window.alert("keyword removed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.CATEGORY_ADDED)) {
            Window.alert("category added - " + event.getType());
        } else if (event.equals(HasDocumentEvent.CATEGORY_REMOVED)) {
            Window.alert("category removed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.TAB_CHANGED)) {
            Window.alert("tab changed - " + event.getType() + " - actual tab " + TabD
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
        } else if (event.equals(HasDocumentEvent.PANEL_RESIZED)) {
            Window.alert("panel resized - " + event.getType());
        } else if (event.equals(HasDocumentEvent.SECURITY_CHANGED)) {
            Window.alert("security changed - " + event.getType());
        } else if (event.equals(HasDocumentEvent.NOTE_ADDED)) {
            Window.alert("note added - " + event.getType());
        }
    }
}
}
```

TabDocumentComunicator

Use to retrieve tab document data.

Methods

method	return	type	Description
getSelectedTab()	***int***	***read***	*Get the selected tab id.*
getDocument()	***GWTDocument***	***read***	*Retrieve the GWT Document object from selected document of the user interface.*
addKeyword(String keyword)	***void***	***write***	*Add a keyword to the selected document of the user interface.*
removeKeyword(String keyword)	***void***	***write***	*Delete a keyword to the selected document of the user interface.*
addCategory(GWTFolder category)	***void***	***write***	*Add a category to the selected document of the user interface.*
removeCategory(String uuid)	***void***	***write***	*Delete a category to the selected document of the user interface.*
getKeywords()	***Collection<String>***	***read***	*Retrieve a list of keywords from selected document of the user interface.*
getNotes()	***Collection<GWTNote>***	***read***	*Retrieve a list of notes from selected document of the user interface.*
isVisibleButton()	***boolean***	***read***	*Return true when buttons are enabled on this view.*
setRefreshingStyle()	***void***	***write***	*Change panel style to refreshing value.*
unsetRefreshingStyle()	***void***	***write***	*Change panel style to default value.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

refreshPreviewDocument()	***void***	***read***	*Force refreshing preview tab.*
isWidgetExtensionVisible(Widget widget)	***boolean***	***read***	*Return true when the widget is visible in document tabs.*

Sample

```
int selectedTab = TabDocumentComunicator.getSelectedTab();
if (selectedTab==0) {
  Window.alert("First tab enabled");
} else {
  Window.alert("Other tab");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabFolderExtension

Add a new tab at bottom right (usually there are shown properties, notes and security tabs) in folder tab section.

Methods

method	Description
getTabText()	Used by OpenKM to get the tab text.
set(GWTFolder folder)	Each time some folder is selected (at tree or browser) is executed this method by OpenKM to refreshing folder tab panel information.
setVisibleButtons(boolean visible)	Show or hide buttons (for example in trash view usually edit button should not be visible).

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTFolder;
import com.openkm.frontend.client.extension.widget.tabfolder.TabFolderExtension;

public class TabFolderExample extends TabFolderExtension {

    VerticalPanel vPanel;

    public TabFolderExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTFolder doc) {
    }

    @Override
    public void setVisibleButtons(boolean visible) {
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FolderHandlerExtension

Any extension that implements FolderHandlerExtension will be looking all events triggered from folder ta.

Methods

method	Description
onChange(FolderEventConstant event)	*Each time it'll be some new folder event the method onChange it'll be fired by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTFolder;
import com.openkm.frontend.client.extension.event.HasFolderEvent;
import com.openkm.frontend.client.extension.event.HasFolderEvent.FolderEventConstant;
import com.openkm.frontend.client.extension.event.handler.FolderHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabfolder.TabFolderExtension;

public class TabFolderExample extends TabFolderExtension implements FolderHandlerExte

    VerticalPanel vPanel;

    public TabFolderExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTFolder doc) {
    }

    @Override
    public void setVisibleButtons(boolean visible) {
    }

    @Override
    public void onChange(FolderEventConstant event) {
        if (event.equals(HasFolderEvent.TAB_CHANGED)) {
            Window.alert("tab changed - " + event.getType());
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabFolder Comunicator

Use to retrieve tab folder data.

Methods

method	return	type	Description
getSelectedTab()	***int***	***read***	*Gets the selected tab id.*
getFolder()	***GWTFolder***	***read***	*Retrieves the GWT folder object from the selected folder of the user interface.*
isVisibleButton()	***boolean***	***read***	*Returns true when buttons are enabled on this view.*
addKeyword(String keyword)	***void***	***write***	*Add a keyword to the selected folder of the user interface.*
removeKeyword(String keyword)	***void***	***write***	*Delete a keyword to the selected folder of the user interface.*
addCategory(GWTFolder category)	***void***	***write***	*Add a category to the selected folder of the user interface.*
removeCategory(String uuid)	***void***	***write***	*Delete a category to the selected folder of the user interface.*
getKeywords()	***Collection<String>***	***read***	*Retrieve a list of keywords from selected folder of the user interface.*
getNotes()	***Collection<GWTNote>***	***read***	*Retrieve a list of notes from selected folder of the user interface.*
isWidgetExtensionVisible(Widget widget)	***boolean***	***read***	*Returns true when the widget is visible in folder tabs.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Sample

```
int selectedTab = TabFolderComunicator.getSelectedTab();
if (selectedTab==0) {
  Window.alert("First tab enabled");
} else {
  Window.alert("Other tab");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabMail Extension

Adds a new tab at bottom right (usually there are shown properties, notes, security and preview tabs) in mail tab section.

Methods

method	Description
getTabText()	*Used by OpenKM to get the tab text.*
set(GWTMail mail)	*Each time there's a mail selected (in a browser) it is executed this method by OpenKM to refresh mail tab panel information.*
setVisibleButtons(boolean visible)	*Shows or hide buttons (for example in trash view usually edit button should not be visible).*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTMail;
import com.openkm.frontend.client.extension.widget.tabmail.TabMailExtension;

public class TabMailExample extends TabMailExtension {

    VerticalPanel vPanel;

    public TabMailExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTMail mail) {
        // TODO Auto-generated method stub
    }

    @Override
    public void setVisibleButtons(boolean visible) {
        // TODO Auto-generated method stub
```

```
        }

    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

MailHandlerExtension

Any extension that implements MailHandlerExtension will be looking all events triggered from mail ta.

Methods

method	Description
onChange(MailEventConstant event)	*Each time it'll be some new mail event the method onChange it'll be fired by OpenKM.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTMail;
import com.openkm.frontend.client.extension.event.HasMailEvent;
import com.openkm.frontend.client.extension.event.HasMailEvent.MailEventConstant;
import com.openkm.frontend.client.extension.event.handler.MailHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabmail.TabMailExtension;

public class TabMailExample extends TabMailExtension implements MailHandlerExtension

    VerticalPanel vPanel;

    public TabMailExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTMail mail) {
        // TODO Auto-generated method stub
    }

    @Override
    public void setVisibleButtons(boolean visible) {
        // TODO Auto-generated method stub
    }

    @Override
    public void onChange(MailEventConstant event) {
        if (event.equals(HasMailEvent.TAB_CHANGED)) {
            Window.alert("tab changed - " + event.getType());
        }
```

```
        }

    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabMail Comunicator

Used to retrieve tab mail data.

Methods

method	return	type	Description
getSelectedTab()	***int***	***read***	*Gets the selected tab id.*
getMail()	***GWTMail***	***read***	*Retrieves the GWT mail object from selected mail of the user interface.*
isWidgetExtensionVisible(Widget widget)	***boolean***	***read***	*Returns true when the widget is visible in mail tabs.*

Sample

```
int selectedTab = TabMailComunicator.getSelectedTab();
if (selectedTab == 0) {
  Window.alert("First tab enabled");
} else {
  Window.alert("Other tab");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabRecord Extension

Adds a new tab at bottom right (usually there are shown properties, notes and security tabs) in record tab section.

Methods

method	Description
getTabText()	Used by OpenKM to get the tab text.
set(GWTRecord record)	Each time there's some record selected (in browser) is executed this method by OpenKM to refresh record tab panel information.
setVisibleButtons(boolean visible)	Shows or hides buttons (for example in trash view normally edit button should not be visible).

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTRecord;
import com.openkm.frontend.client.extension.widget.tabrecord.TabRecordExtension;

public class TabRecordExample extends TabRecordExtension {

    VerticalPanel vPanel;

    public TabRecordExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTRecord record) {
        // TODO Auto-generated method stub
    }

    @Override
    public void setVisibleButtons(boolean visible) {
        // TODO Auto-generated method stub
```

```
        }
    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

RecordHandlerExtension

Any extension that implements RecordHandlerExtension will be monitoring all the events triggered from record ta.

Methods

method	*Description*
onChange(RecordEventConstant event)	*Each time it'll be some new Record event the method onChange it'll be triggered by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.openkm.frontend.client.bean.GWTRecord;
import com.openkm.frontend.client.extension.event.HasRecordEvent;
import com.openkm.frontend.client.extension.event.HasRecordEvent.RecordEventConstant;
import com.openkm.frontend.client.extension.event.handler.RecordHandlerExtension;
import com.openkm.frontend.client.extension.widget.tabrecord.TabRecordExtension;

public class TabRecordExample extends TabRecordExtension implements RecordHandlerExte

    VerticalPanel vPanel;

    public TabRecordExample() {
        vPanel = new VerticalPanel();
        vPanel.add(new HTML("hello world"));

        initWidget(vPanel);
    }

    @Override
    public String getTabText() {
        return "New folder tab";
    }

    @Override
    public void set(GWTRecord record) {
        // TODO Auto-generated method stub
    }

    @Override
    public void setVisibleButtons(boolean visible) {
        // TODO Auto-generated method stub
    }

    @Override
    public void onChange(RecordEventConstant event) {
        if (event.equals(HasRecordEvent.TAB_CHANGED)) {
            Window.alert("tab changed - " + event.getType());
```

```
                }
            }
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TabRecord Comunicator

Used to retrieve tab record data.

Methods

method	return	type	Description
getSelectedTab()	***int***	***read***	*Get the selected tab id.*
getRecord()	***GWTRecord***	***read***	*Retrieve the GWT record object from selected record of the user interface.*
isWidgetExtensionVisible(Widget widget)	***boolean***	***read***	*Return true when the widget is visible in record tabs.*

Sample

```
int selectedTab = TabRecordComunicator.getSelectedTab();
if (selectedTab == 0) {
  Window.alert("First tab enabled");
} else {
  Window.alert("Other tab");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ToolBarBox Extension

Adds a new button at dthe ashboard view.

Methods

method	Description
getWidget()	*Returns the associated widget to the tool box that will be showed when it'll is selected.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.HorizontalPanel;
import com.google.gwt.user.client.ui.Image;
import com.google.gwt.user.client.ui.Widget;
import com.openkm.extension.frontend.client.util.OKMExtensionBundleExampleResources;
import com.openkm.frontend.client.extension.widget.toolbar.ToolBarBoxExtension;

public class ToolBarBoxExample {

    public HorizontalPanel hPanel;
    public ToolBarBoxExtension toolBarBoxExtension;

    /**
     * ToolBarBoxExample
     */
    public ToolBarBoxExample() {
        toolBarBoxExtension = new ToolBarBoxExtension(new Image(OKMExtensionBundleExa
                "new toolbar") {
            @Override
            public Widget getWidget() {
                return hPanel;
            }
        };

        hPanel = new HorizontalPanel();
        hPanel.add(new HTML("new toolbarbox example"));
    }

    /**
     * ToolBarBoxExtension
     */
    public ToolBarBoxExtension getToolBarBox() {
        return toolBarBoxExtension;
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

DashboardHandlerExtension

Any extension that implements ToolBarHandlerExtension will be monitoring events triggered from the toolbar.

Methods

method	Description
onChange(DashboardEventConstant event)	*Each time it'll be some new toolbar event the method onChange it'll be fired by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.HTML;
import com.google.gwt.user.client.ui.HorizontalPanel;
import com.google.gwt.user.client.ui.Image;
import com.google.gwt.user.client.ui.Widget;
import com.openkm.extension.frontend.client.util.OKMExtensionBundleExampleResources;
import com.openkm.frontend.client.extension.event.HasDashboardEvent;
import com.openkm.frontend.client.extension.event.HasToolBarEvent.ToolBarEventConstan
import com.openkm.frontend.client.extension.event.handler.ToolBarHandlerExtension;
import com.openkm.frontend.client.extension.widget.toolbar.ToolBarBoxExtension;

public class ToolBarBoxExample implements DashboardHandlerExtension {

    public HorizontalPanel hPanel;
    public ToolBarBoxExtension toolBarBoxExtension;

    /**
     * ToolBarBoxExample
     */
    public ToolBarBoxExample() {
        toolBarBoxExtension = new ToolBarBoxExtension(new Image(OKMExtensionBundleExa
                "new toolbar") {
            @Override
            public Widget getWidget() {
                return hPanel;
            }
        };

        hPanel = new HorizontalPanel();
        hPanel.add(new HTML("new toolbarbox example"));
    }

    /**
     * ToolBarBoxExtension
     */
    public ToolBarBoxExtension getToolBarBox() {
        return toolBarBoxExtension;
    }

    @Override
```

```
    public void onChange(DashboardEventConstant event) {
        if (event.equals(HasDashboardEvent.TOOLBOX_CHANGED)) {
            Window.alert("toolbar changed - " + event.getType());
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ToolBarComunicator

Use it to retrieve toolbar data.

Methods

method	return	type	Description
getToolBarOption()	***ToolBarOption***	***read***	*Retrieve the GWT ToolBarOption object what contains information about enabled or disabled UI features.*
setToolBarOption(ToolBarOption toolBarOption)	***void***	***write***	*Change toolbar option, it enable or disable UI features.*
evaluateShowIcons()	***void***	***read***	*Evaluate toolbar icons based on security.*
getActualNode()	***Object***	***read***	*Retrieve the selected node of the user interface.* *Retrieved object can be cast to:* • *GWTDocument.* • *GWTFolder.* • *GWTMail.* • *GWTRecrod.*
getMainToolBarPanel()	***HorizontalPanel***	***read***	*Retrieve the main widget of the tool bar.*

Sample

```
ToolBarComunicator.evaluateShowIcons();
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ToolBarButton Extension

Adds a new toolbar icon. Usually toolbar icons are OpenKM features like create folder, delete, upload new file, etc.

Methods

method	Description
checkPermissions(GWTNode node, GWTNode nodeParent, int originPanel)	*Checks the button permissions based on the selected node grants to enable or disable the button.*
checkPermissions(GWTDocument doc, GWTNode node)	*Checks the button permissions based on selected document grants to enable or disable the button.*
checkPermissions(GWTMail mail, GWTNode node)	*Checks the button permissions based on the selected folder grants to enable or disable the button.*
checkPermissions(GWTRecord record, GWTNode node);	*Checks the button permissions based on the selected record node grants to enable or disable the button.*
isEnabled()	*Returns a boolean value indicating if the button is enabled or disabled.*
enable(boolean enable)	*Enables or disables button.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.Image;
import com.openkm.extension.frontend.client.util.OKMExtensionBundleExampleResources;
import com.openkm.frontend.client.bean.GWTDocument;
import com.openkm.frontend.client.bean.GWTMail;
import com.openkm.frontend.client.bean.GWTNode;
import com.openkm.frontend.client.bean.GWTRecord;
import com.openkm.frontend.client.extension.widget.toolbar.ToolBarButtonExtension;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
public class ToolBarButtonExample {

    ToolBarButton button;
    String title = "title";
    String lang = "en-GB";

    public ToolBarButtonExample() {
        button = new ToolBarButton(new Image(OKMExtensionBundleExampleResources.INSTA
            @Override
            public void onClick(ClickEvent event) {
                Window.alert("make some operation");
            }
        });
    }

    /**
     * ToolBarButtonExtension
     */
    public ToolBarButtonExtension getButton() {
        return button;
    }

    private class ToolBarButton extends ToolBarButtonExtension {

        public ToolBarButton(Image image, String title, ClickHandler handler) {
            super(image, title, handler);
        }

        @Override
        public void checkPermissions(GWTNode node, GWTNode nodeParent, int originPane
        }

        @Override
        public void checkPermissions(GWTDocument doc, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTMail mail, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTRecord record, GWTNode node) {
        }

        @Override
        public void enable(boolean enable) {
        }

        @Override
        public boolean isEnabled() {
            return false;
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ToolBarHandler Extension

Any extension that implements ToolBarHandlerExtension will be monitoring events triggered from the toolbar.

Methods

method	Description
onChange(ToolBarEventConstant event)	*Each time it'll be some new toolbar event the method onChange it'll be fired by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.Image;
import com.openkm.extension.frontend.client.util.OKMExtensionBundleExampleResources;
import com.openkm.frontend.client.bean.GWTDocument;
import com.openkm.frontend.client.bean.GWTMail;
import com.openkm.frontend.client.bean.GWTNode;
import com.openkm.frontend.client.bean.GWTRecord;
import com.openkm.frontend.client.extension.event.HasToolBarEvent;
import com.openkm.frontend.client.extension.event.HasToolBarEvent.ToolBarEventConstan
import com.openkm.frontend.client.extension.event.handler.ToolBarHandlerExtension;
import com.openkm.frontend.client.extension.widget.toolbar.ToolBarButtonExtension;

public class ToolBarButtonExample {

    ToolBarButton button;
    String title = "title";
    String lang = "en-GB";

    public ToolBarButtonExample() {
        button = new ToolBarButton(new Image(OKMExtensionBundleExampleResources.INSTA
            @Override
            public void onClick(ClickEvent event) {
                Window.alert("make some operation");
            }
        });
    }

    /**
     * ToolBarButtonExtension
     */
    public ToolBarButtonExtension getButton() {
        return button;
    }

    private class ToolBarButton extends ToolBarButtonExtension implements ToolBarHand

        public ToolBarButton(Image image, String title, ClickHandler handler) {
            super(image, title, handler);
```

```
        }

        @Override
        public void checkPermissions(GWTNode node, GWTNode nodeParent, int originPane
        }

        @Override
        public void checkPermissions(GWTDocument doc, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTMail mail, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTRecord record, GWTNode node) {
        }

        @Override
        public void enable(boolean enable) {
        }

        @Override
        public boolean isEnabled() {
            return false;
        }

        @Override
        public void onChange(ToolBarEventConstant event) {
            if (event.equals(HasToolBarEvent.EXECUTE_ADD_DOCUMENT)) {
                Window.alert("executed add document - " + event.getType());
            }
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

MenuBarExtension

Add new menu bar.

Contructor

method	*Description*
MenuBarExtension()	*Create a new menu bar or submenu.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Command;
import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.widget.menu.MenuBarExtension;
import com.openkm.frontend.client.extension.widget.menu.MenuItemExtension;

public class MainMenuExample {

    private MenuItemExtension exampleMenu;
    private MenuBarExtension subMenuExample;
    private MenuBarExtension subMenuExample2;
    private MenuItemExtension subMenuItem;
    private MenuItemExtension option1;
    private MenuItemExtension option2;
    private MenuItemExtension option3;
    private MenuItemExtension option4;

    /**
     * MainMenuExample
     */
    public MainMenuExample() {
        // All menu items
        option1 = new MenuItemExtension("img/box.png", "Option 1", option1Action);
        option2 = new MenuItemExtension("img/box.png", "Option 2", option2Action);
        option3 = new MenuItemExtension("img/box.png", "Option 3", option3Action);
        option4 = new MenuItemExtension("img/box.png", "Option 4", option4Action);

        // Secondary submenu
        subMenuExample2 = new MenuBarExtension();
        subMenuExample2.addItem(option3);
        subMenuExample2.addItem(option4);
        subMenuItem = new MenuItemExtension("img/box.png", "Sub menu", subMenuExample

        // Principal submenu
        subMenuExample = new MenuBarExtension();
        subMenuExample.addItem(option1);
        subMenuExample.addItem(option2);
        subMenuExample.addItem(subMenuItem);

        // Principal menuitem
        exampleMenu = new MenuItemExtension("New Menu", subMenuExample); // is not a
    }
```

```
    public MenuItemExtension getNewMenu() {
        return exampleMenu;
    }

    /**
     * option1Action
     */
    Command option1Action = new Command() {
        public void execute() {
            Window.alert("option1 action");
        }
    };

    /**
     * option2Action
     */
    Command option2Action = new Command() {
        public void execute() {
            Window.alert("option2 action");
        }
    };

    /**
     * option3Action
     */
    Command option3Action = new Command() {
        public void execute() {
            Window.alert("option3 action");
        }
    };

    /**
     * option4Action
     */
    Command option4Action = new Command() {
        public void execute() {
            Window.alert("option4 action");
        }
    };
}
```

MenuItemExtension

Add new menu option to the menu bar.

Constructors

method	*Description*
MenuItemExtension(String imageURL, String text, Command cmd)	*Default menuItem.*
MenuItemExtension(String imageURL, String text, MenuBar menuBar)	*A menuitem that has a menu bar into. Usually used for a sub menu entry.*
MenuItemExtension(String text, MenuBar menuBar)	*Usually used to define the main menuItem. This is the root menu that has menu children options.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Command;
import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.widget.menu.MenuBarExtension;
import com.openkm.frontend.client.extension.widget.menu.MenuItemExtension;

public class MainMenuExample {

    private MenuItemExtension exampleMenu;
    private MenuBarExtension subMenuExample;
    private MenuBarExtension subMenuExample2;
    private MenuItemExtension subMenuItem;
    private MenuItemExtension option1;
    private MenuItemExtension option2;
    private MenuItemExtension option3;
    private MenuItemExtension option4;

    /**
     * MainMenuExample
     */
    public MainMenuExample() {
        // All menu items
        option1 = new MenuItemExtension("img/box.png", "Option 1", option1Action);
        option2 = new MenuItemExtension("img/box.png", "Option 2", option2Action);
        option3 = new MenuItemExtension("img/box.png", "Option 3", option3Action);
        option4 = new MenuItemExtension("img/box.png", "Option 4", option4Action);

        // Secondary submenu
        subMenuExample2 = new MenuBarExtension();
```

```
        subMenuExample2.addItem(option3);
        subMenuExample2.addItem(option4);
        subMenuItem = new MenuItemExtension("img/box.png", "Sub menu", subMenuExample

        // Principal submenu
        subMenuExample = new MenuBarExtension();
        subMenuExample.addItem(option1);
        subMenuExample.addItem(option2);
        subMenuExample.addItem(subMenuItem);

        // Principal menuitem
        exampleMenu = new MenuItemExtension("New Menu", subMenuExample); // is not a
    }

    public MenuItemExtension getNewMenu() {
        return exampleMenu;
    }

    /**
     * option1Action
     */
    Command option1Action = new Command() {
        public void execute() {
            Window.alert("option1 action");
        }
    };

    /**
     * option2Action
     */
    Command option2Action = new Command() {
        public void execute() {
            Window.alert("option2 action");
        }
    };

    /**
     * option3Action
     */
    Command option3Action = new Command() {
        public void execute() {
            Window.alert("option3 action");
        }
    };

    /**
     * option4Action
     */
    Command option4Action = new Command() {
        public void execute() {
            Window.alert("option4 action");
        }
    };
}
```

General HandlerExtension stuff

DashboardHandlerExtension

Any extension that implements DashboardHandlerExtension will be watching all events fired from Dashboard.

Methods

method	Description
onChange(DashboardEventConstant event)	Each time it'll be a new dashboard event the method onChange it'll be fired by OpenKM.

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.event.HasDashboardEvent.DashboardEventCon
import com.openkm.frontend.client.extension.event.handler.DashboardHandlerExtension;

public class HandlerTest implements DashboardHandlerExtension {

    @Override
    public void onChange(DashboardEventConstant event) {
        Window.alert("dashboard event fired");
    }

}
```

LanguageHandlerExtension

Any extension that implements LanguageHandlerExtension will be watching all events from change Language.

Methods

method	Description
onChange(LanguageEventConstant event)	Each time it'll be a new language event the method onChange it'll be fired by OpenKM.

Sample

```
package com.openkm.extension.frontend.client;
```

```
import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.event.HasLanguageEvent.LanguageEventConst
import com.openkm.frontend.client.extension.event.handler.LanguageHandlerExtension;

public class HandlerTest implements LanguageHandlerExtension {

    @Override
    public void onChange(LanguageEventConstant event) {
        Window.alert("language event fired");
    }

}
```

NavigatorHandlerExtension

Any extension that implements NavigatorHandlerExtension will be watching all events fired from navigator stack panel.

Methods

method	Description
onChange(NavigatorEventConstant event)	*Each time it'll be a new stack panel event the method onChange it'll be fired by OpenKM.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.event.HasNavigatorEvent.NavigatorEventCon
import com.openkm.frontend.client.extension.event.handler.NavigatorHandlerExtension;

public class HandlerTest implements NavigatorHandlerExtension {

    @Override
    public void onChange(NavigatorEventConstant event) {
        Window.alert("navigator event fired");
    }

}
```

PropertyGroupHandlerExtension

Any extension that implements PropertyGroupHandlerExtension will be watching all events fired from metadata group tab.

Methods

method	Description

onChange(PropertyGroupEventConstant event)	*Each time it'll be a new property group (metadata) event the method onChange it'll be fired by OpenKM.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.event.HasPropertyGroupEvent;
import com.openkm.frontend.client.extension.event.HasPropertyGroupEvent.PropertyGroup
import com.openkm.frontend.client.extension.event.handler.PropertyGroupHandlerExtensi

public class HandlerTest implements PropertyGroupHandlerExtension {

    @Override
    public void onChange(PropertyGroupEventConstant event) {
        if (event.equals(HasPropertyGroupEvent.PROPERTYGROUP_CHANGED)) {
            Window.alert("propertyGroup changed - " + event.getType());
        } else {
            Window.alert("other property group event - " + event.getType());
        }
    }

}
```

WidgetHandlerExtension

Any extension that implements WidgetHandlerExtension will be watching all events fired from widgets.

Methods

method	*Description*
onChange(WidgetEventConstant event)	*Each time it'll be a new widget event the method onChange it'll be fired by OpenKM .*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.event.HasWidgetEvent;
import com.openkm.frontend.client.extension.event.HasWidgetEvent.WidgetEventConstant;
import com.openkm.frontend.client.extension.event.handler.WidgetHandlerExtension;

public class HandlerTest implements WidgetHandlerExtension {

    @Override
    public void onChange(WidgetEventConstant event) {
        if (event.equals(HasWidgetEvent.FINISH_STARTUP)) {
            Window.alert("widget event fired");
        }
```

```
    }

}
```

WorkspaceUserPropertiesHandlerExtension

Any extension that implements WorkspaceUserHandlerExtension will be watching all events fired from Workspace user.

Methods

method	*Description*
workspaceUserLoaded()	*It is called the first time workspace user properties are loaded.*

Sample

```
package com.openkm.extension.frontend.client;

import com.google.gwt.user.client.Window;
import com.openkm.frontend.client.extension.communicator.GeneralCommunicator;
import com.openkm.frontend.client.extension.event.handler.WorkspaceUserPropertiesHand

public class HandlerTest implements WorkspaceUserPropertiesHandlerExtension {

    @Override
    public void workspaceUserLoaded() {
        Window.alert("workspace user properties loaded can call GeneralCommunicator t
        if (GeneralCommunicator.getWorkspace() != null) {
            Window.alert("workspace loaded");
        }
    }

}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

General Comunicator stuff

GeneralComunicator

Use it to retrieve general data.

Methods

method	return	type	Description
refreshUI()	**void**	**read**	*Refresh user interface.*
getToolBarOption()	**ToolBarOption**	**read**	*Retrive ToolbarOption object.* *The variables into ToolbarOption shown OpenKM features enabled or disabled by security.*
getLang()	**String**	**read**	*Get user interface language code.*
i18nExtension(String property)	**String**	**read**	*Retrieve extension translation.*
i18n(String property)	**String**	**read**	*Retrieve frontend translation.*
downloadDocument(boolean checkout)	**void**	**read**	*Download selected document.* *When checkout value is "**true**" the document will be marked as "**edition**".*
downloadDocumentPdf()	**void**	**read**	*Download selected document as pdf.*
extensionCallOwnDownload(String url)	**void**	**read**	*Used to call*
downloadFileByUUID(String uuid,	**void**	**read**	*Download file with parameters.*

String params)			*Allowed parameters:* • *The "**checkout**" parameter mark document as "edit".* • *The "**inline**" parameter to download as attachment, for example to downloading images.* • *The "**ver**" parameter to download specific document version.*
downloadFile(String path, String params)	***void***	***read***	*Download file with parameters.* *Deprecated method in favritesr "**downloadFileByUUID**" method.*
setStatus(String msg)	***void***	***void***	*On bottom right change status message.*
resetStatus()	***void***	***read***	*Reset bottom right status bar.*
showError(String callback, Throwable caught)	***void***	***read***	*Show popup error with some message.*
logout()	***void***	***read***	*Logout from OpenKM user interface.*
getUserRoleList()	***List<String>***	***read***	*Retrieve a list with the roles of the user.*
getUser()	***String***	***read***	*Retrieve userId.*
openPath(String path, String docPath)	***void***	***read***	*Open node into openkm.* *Parameters:*

			• *Parameter "**path**" should be a valid "**folder path**".* • *Parameter "**docPath**" should be empty or a valid "**document path**".* *To open a single folder path the parameter "**docPath**" must be empty.*
openAllFolderPath(String path, String docPath)	***void***	***read***	*Open all Folder into openkm.* *Parameters:* • *Parameter "**path**" should be a valid "**folder path**".* • *Parameter "**docPath**" should be empty or a valid "**document path**".* *To open a single folder path the parameter "**docPath**" must be empty.*
getAppContext()	***String***	***read***	*Retrieve the application context.*
showNextWizard()	***void***	***read***	*Show next wizard.* *Wizards are shown after document uploading.*
isDigitalSignature()	***boolean***	***read***	*When true indicate digital signature extension is enabled.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getDocumentToSign()	***GWTDocument***	***read***	*Retrieve a document object to be signed.*
getSessionId()	***String***	***read***	*Retrive a session.*
getWorkspace()	***GWTWorkspace***	***read***	*Retrieve the GWT Workspace object.* *Into the object is set information about UI features - enabled or disabled - among others.*
enableKeyShorcuts()	***void***	***read***	*Enable UI shortcuts.*
disableKeyShorcuts()	***void***	***void***	*Disable UI shortcuts.*
openPathByUuid(String uuid)	***void***	***void***	*Open node into openkm.*
getFolderIcon(GWTFolder fld)	***String***	***void***	*Retrieve folder icon image path.*
get()	***Main***	***void***	*Retrieve Main object from singleton pattern.* *Main class is the principal frontend class which offers full access to any frontend UI method.*

Sample

```
// Call to refreshing user interface
GeneralComunicator.refreshUI();
```

DashboardComunicator

Use it to retrieve tab dashboard data.

Methods

method	*return type*	*Description*
getUserSubscribedDocuments()	***void*** ***read***	*Refresh user subscribed documents list.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getUserSubscribedFolders()	***void***	***read***	*Refresh user subscribed folders list.*
refreshAllSearchs()	***void***	***read***	*Refresh search list.*
showToolBoxExtension(ToolBarBoxExtension extension)	***void***	***read***	*Change selected dashboard button.*
getUserSearchs(boolean refresh)	***void***	***read***	*Refresh user search list.* *When refresh boolean value is true, also refresh search results at dashboard news view.*
isWidgetExtensionVisible(Widget widget)	***boolean***	***read***	*When true indicate widget extension is enabled.*

Sample

```
// Call to refreshing subscribed documents
DashboardComunicator.getUserSubscribedDocuments();
```

FileBrowserComunicator

Use it to retrieve file browser data.

Methods

method	*return*	*type*	*Description*
isDocumentSelected()	***boolean***	***read***	*When document is selected it returns "true" value.*
isFolderSelected()	***boolean***	***read***	*When folder is selected it returns "true" value.*
isMailSelected()	***boolean***	***read***	*When mail is selected it returns "true" value.*
isRecordSelected()	***boolean***	***read***	*When record is selected it returns "true" value.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getDocument()	***GWTDocument***	***read***	*Retrieve a GWT document object of the selected document.*
getFolder()	***GWTFolder***	***read***	*Retrieve a GWT folder object of the selected folder.*
getMail()	***GWTMail***	***read***	*Retrieve a GWT mail object of the selected mail.*
getRecord()	***GWTRecord***	***read***	*Retrieve a GWT record object of the selected record.*
isPanelSelected()	***boolean***	***read***	*When file browser panel is selected return "true" value.*
refreshOnlyFileBrowser()	***void***	***read***	*Refresh only file browser widget view.*
isMassive()	***boolean***	***read***	*When true indicate massive is enabled.*
getAllSelectedUUIDs()	***List<String>***	***read***	*Retrieve a list with the selected uuids.*

Sample

```
if (FileBrowserComunicator.isDocumentSelected()) {
    Window.alert("document selected");
} else {
    Window.alert("document not selected");
}
```

NavigatorComunicator

Use it to retrieve navigator data.

The navigator panel contains stack panel with folder trees into (taxonomy, categories etc...).

Methods

method	*return*	*type*	*Description*
getStackIndex()	***int***	***read***	*Get the stack index id.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

isTaxonomyShown()	***boolean***	***read***	*When true, indicate the taxonomy stack is shown.*
isCategoriesShown()	***boolean***	***read***	*When true, indicate the categories stack is shown.*
isMetadataShown()	***boolean***	***read***	*When true, indicate the metadata stack is shown.*
isThesaurusShown()	***boolean***	***read***	*When true, indicate the thesaurus stack is shown.*
isTemplatesShown	***boolean***	***read***	*When true, indicate the templates stack is shown.*
isPersonalShown	***boolean***	***read***	*When true, indicate the personal stack is shown.*
isMailShown	***boolean***	***read***	*When true, indicate the mail stack is shown.*
isTrashShown	***boolean***	***read***	*When true, indicate the trash stack is shown.*
getFolder()	***GWTFolder***	***read***	*Retrieve the selected folder object from enabled tree view.* *We suggest using getNode() method rather this one.*
getActualPath()	***String***	***read***	*Retrieve the path of the selected folder from enabled tree view.*
getRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:root**".*
getCategoriesRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:categories**".*
getMetadataRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:metadata**".*
getThesaurusRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:thesaurus**".*
getTemplatesRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:templates**".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getPersonalRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:personal**".*
getMailRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:mail**".*
getTrashRootFolder()	***GWTFolder***	***read***	*Retrieve the GWT folder object of node "**/okm:trash**".*
isPanelSelected()	***boolean***	***read***	*When "**true**" indicate the navigator panel is the selected node.*
getNode()	***GWTNode***	***read***	*Retrieve the selected node object from enabled tree view.* *The selected node can be folder or a record type.*

Sample

```
int stack = NavigatorComunicator.getStackIndex();
if (stack==0) {
  Window.alert("First stack enabled");
} else {
  Window.alert("Other stack");
}
```

SearchComunicator

Use it to retrieve search data.

Methods

method	*return*	*type*	*Description*
setSavedSearch(GWTQueryParams params)	***void***	***write***	*Execute a saved search.*
getAllSearchs()	***void***	***read***	*Refresh saved search panel.*
getUserSearchs()	***void***	***read***	*Refresh user saved search panel.*
getSelectedRowSearchSaved()	***int***	***read***	*Retrieve the selected row of*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

			saved search panel.
getSelectedRowUserNews()	***int***	***read***	*Retrieve the selected row of user saved search panel.*
getSavedSearch()	***GWTQueryParams***	***read***	*Retrieve the GWT QueryParams object of the selected saved search.*
getSavedUserNews()	***GWTQueryParams***	***read***	*Retrieve the GWT QueryParams object of the selected user saved search.*
showFolderSelectPopup(FindFolderHandler findFolderHandler)	***void***	***read***	*Register a handler for the find folder popup.*
showDocumentSelectPopup(FindDocumentHandler findDocumentHandler, int typeSearch)	***void***	***read***	*Register a handler for the find document popup.*
showMailSelectPopup(FindMailHandler findMailHandler)	***void***	***read***	*Register a handler for the find mail popup.*
showRecordSelectPopup(FindRecordHandler findRecordHandler)	***void***	***read***	*Register a handler for the find record popup.*

Sample

```
// Call to refreshing subscribed documents
GWTQueryParams params = new GWTQueryParams();
params.setName("test");
SearchComunicator.setSavedSearch(params);
```

UtilComunicator

General purpose methods.

Methods

method	*return type*	*Description*

formatSize(double size)	***String***	***read***	*Convert document size in bytes to human value (Kb, Mb, GB etc..).*
createHeaderHTML(String imageURL, String caption)	***String***	***read***	*Retrieve a stack header HTML code to be used into GWT frontend stack widget.*
imageItemHTML(String imageUrl, String title, String align)	***String***	***read***	*Retrieve image HTML code to be used into GWT frontend widget.*
menuHTML(String imageUrl, String text)	***String***	***read***	*Retrieve menu HTML code to be used into GWT frontend menu element.*
menuHTMLWithoutIcon(String text)	***String***	***read***	*Retrieve menu HTML code to be used into GWT frontend menu element.*
imageItemHTML(String imageUrl)	***String***	***read***	*Retrieve image HTML code to be used into GWT frontend widget.*
getTextAsBoldHTML(String text, boolean mark)	***String***	***read***	*Retrieve text as HTML bold one.*
getUserAgent()	***String***	***read***	*Retrieve user agent of the browser.*
getName(String path)	***String***	***read***	*Retrieve the name of a path.* *The name of the "/okm:root/temp/doc1.pdf" is "doc1.pdf".*
getParent(String path)	***String***	***read***	*Retrieve the parent path of a path.* *The parent of the "/okm:root/temp/doc1.pdf" is "/okm:root/temp".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

mimeImageHTML(String mime)	***String***	***read***	*Retrieve the url of the MIME TYPE icon.*
hSpace(String width)	***HTML***	***read***	*Retrieve a GWT HTML horizontal separator.*
vSpace(String height)	***HTML***	***read***	*Retrieve a GWT HTML vertical separator.*
isSearchableKey(KeyUpEvent event)	***boolean***	***read***	*Evaluate if the key pressed is not into navigation, arrows or modifier keys.* *Navigation keys: HOME, END, PAGEUP, PAGEDOWN.* *Modifier keys: SHIFT, ALT, CTRL, DOWN.* *Arrow keys: LEFT, RIGHT, UP, DOWN.*

Sample

```
// Call to formatting the document size ( gb, mb etc... )
UtilComunicator.formatSize(15);
```

UI constants

UIMenuConstants

UIMenuConstant class set contants to be used in OpenKM extensions, for example to set new menu options on some existing OpenKM menu.

*When menu location is not set by default is used value **NEW_MENU**. That means it'll be created as new menu into main menu desktop view.*

Sample

Add new submenu in tools menu:

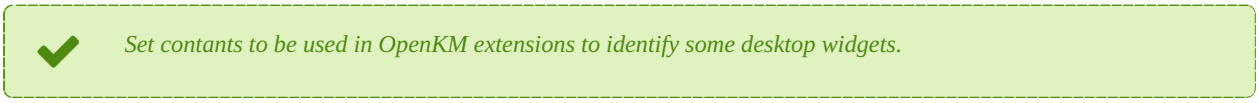
```
public class SubMenuMessage {
 private MenuItemExtension messageMenu;
 private MenuBarExtension subMenuMessage;
 private MenuItemExtension sendNewMessage;

 /**
  * SubMenuMessage
  */
 public SubMenuMessage() {
  // All menu items
  sendNewMessage = new MenuItemExtension("img/icon/actions/new_message.png", "New mes

  // Principal submenu
  subMenuMessage = new MenuBarExtension();
  subMenuMessage.addItem(sendNewMessage);
  messageMenu = new MenuItemExtension("img/icon/actions/message.png", "Message", subM
  messageMenu.setMenuLocation(UIMenuConstants.MAIN_MENU_TOOLS);
 }

         /**
  * @return
  */
 public MenuItemExtension getMenu() {
  return messageMenu;
 }

 /**
  * option1Action
  */
 Command sendMessage = new Command() {
  public void execute() {
   Window.alert("some action");
  }
 };
}
```

UIDashboardConstants

Set contants to be used in OpenKM extensions to identify some widgets on dashboard.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

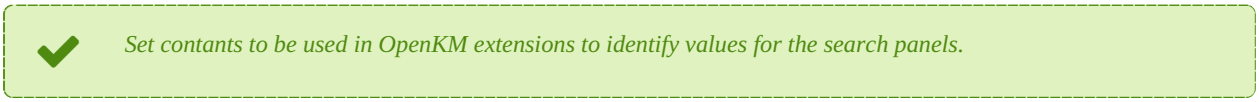
UIDesktopConstants



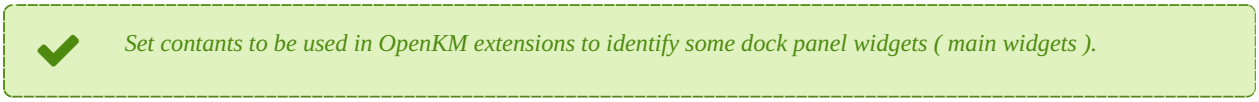
Sample

```
if (NavigatorComunicator.getStackIndex()==UIDesktopConstants.NAVIGATOR_TAXONOMY) {
    Window.alert("Taxonomy selected");
} else {
    Window.alert("Other navigator panel selected");
}
```

UIDialogConstants



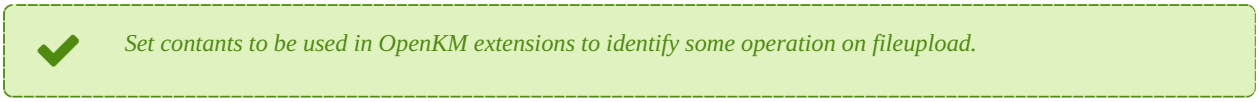
UIDockPanelConstants



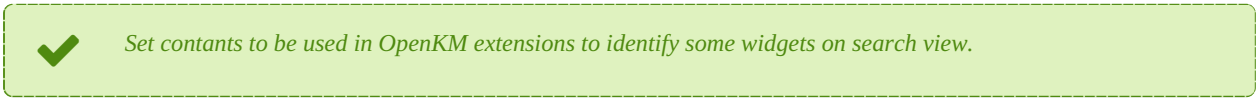
Sample

```
WorkspaceComunicator.changeSelectedTab(UIDockPanelConstants.SEARCH);
```

UIFileUploadConstants



UISearchConstants



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Frontend UI extensions stuff

Internatinalization

Register new translations from Database query.

```
INSERT INTO OKM_TRANSLATION (TR_MODULE, TR_KEY, TR_TEXT, TR_LANGUAGE) VALUES ('extens
INSERT INTO OKM_TRANSLATION (TR_MODULE, TR_KEY, TR_TEXT, TR_LANGUAGE) VALUES ('extens
```

Use in your code:

```
GeneralComunicator.i18nExtension("new_extension.button.add");
```

Also can be used default OpenKM translations:

```
GeneralComunicator.i18n("button.close");
```

Sample

> *Remember any extension internationalized must implements **LanguageHandlerExtension**. Each time the user interface change language this handler is fired.*

```
package com.openkm.extension.frontend.client;

import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.Image;
import com.openkm.extension.frontend.client.util.OKMExtensionBundleExampleResources;
import com.openkm.frontend.client.bean.GWTDocument;
import com.openkm.frontend.client.bean.GWTMail;
import com.openkm.frontend.client.bean.GWTNode;
import com.openkm.frontend.client.bean.GWTRecord;
import com.openkm.frontend.client.extension.event.HasLanguageEvent;
import com.openkm.frontend.client.extension.event.HasLanguageEvent.LanguageEventConst
import com.openkm.frontend.client.extension.event.handler.LanguageHandlerExtension;
import com.openkm.frontend.client.extension.widget.toolbar.ToolBarButtonExtension;

public class ToolBarButtonExample {

    ToolBarButton button;
    String title = "title";
    String lang = "en-GB";

    public ToolBarButtonExample() {
        button = new ToolBarButton(new Image(OKMExtensionBundleExampleResources.INSTA
            @Override
            public void onClick(ClickEvent event) {
                Window.alert("make some operation");
            }
        });
    }
```

```
    /**
     * ToolBarButtonExtension
     */
    public ToolBarButtonExtension getButton() {
        return button;
    }

    /**
     * ToolBarButton
     *
     * @author jllort
     */
    private class ToolBarButton extends ToolBarButtonExtension implements LanguageHan

        public ToolBarButton(Image image, String title, ClickHandler handler) {
            super(image, title, handler);
        }

        @Override
        public void checkPermissions(GWTNode node, GWTNode nodeParent, int originPane
        }

        @Override
        public void checkPermissions(GWTDocument doc, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTMail mail, GWTNode node) {
        }

        @Override
        public void checkPermissions(GWTRecord record, GWTNode node) {
        }

        @Override
        public void enable(boolean enable) {
        }

        @Override
        public boolean isEnabled() {
            return false;
        }

        @Override
        public void onChange(LanguageEventConstant event) {
            if (event.equals(HasLanguageEvent.LANGUAGE_CHANGED)) {
                Window.alert("language changed");
            }
        }
    }
}
```

Registering RPC (Remote Process Call) service

Servlets are used to do GWT RPC (Remote Process Calls). In order to registering a new Servlet file (web.xml) must be modified and OpenKM re-compiled.

*Into **web.xml** (src/main/webapp/WEB-INF) must be added the servlet declaration:*

```
<servlet>
    <servlet-name>SomeServletName</servlet-name>
```

```
    <servlet-class>SomeJavaClassName</servlet-class>
</servlet>

<servlet-mapping>
    <servlet-name>SomeServletName</servlet-name>
    <url-pattern>/SomeServletName</url-pattern>
</servlet-mapping>
```

RPCService class defines all RPC constants service that can be used from GWT:

```
ServiceDefTarget endPoint = (ServiceDefTarget) messageService;
endPoint.setServiceEntryPoint(RPCService.MessageService);
messageService.deleteSent(1, new AsyncCallback<Object>() {
        @Override
 public void onSuccess(Object result) {
 }

 @Override
 public void onFailure(Throwable caught) {
 }
});
```

More information at http://www.gwtproject.org/doc/latest/tutorial/RPC.html.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Complete frontend extension sample

This sample shows how to quickly extend OpenKM features with JSP, HTML, CSS and JavaScript technologies using OpenKM Java API and how to use JavaScript functions of the exposed frontend JAVA methods.

Download the extension sample code complete_extension_sample.zip.

Description

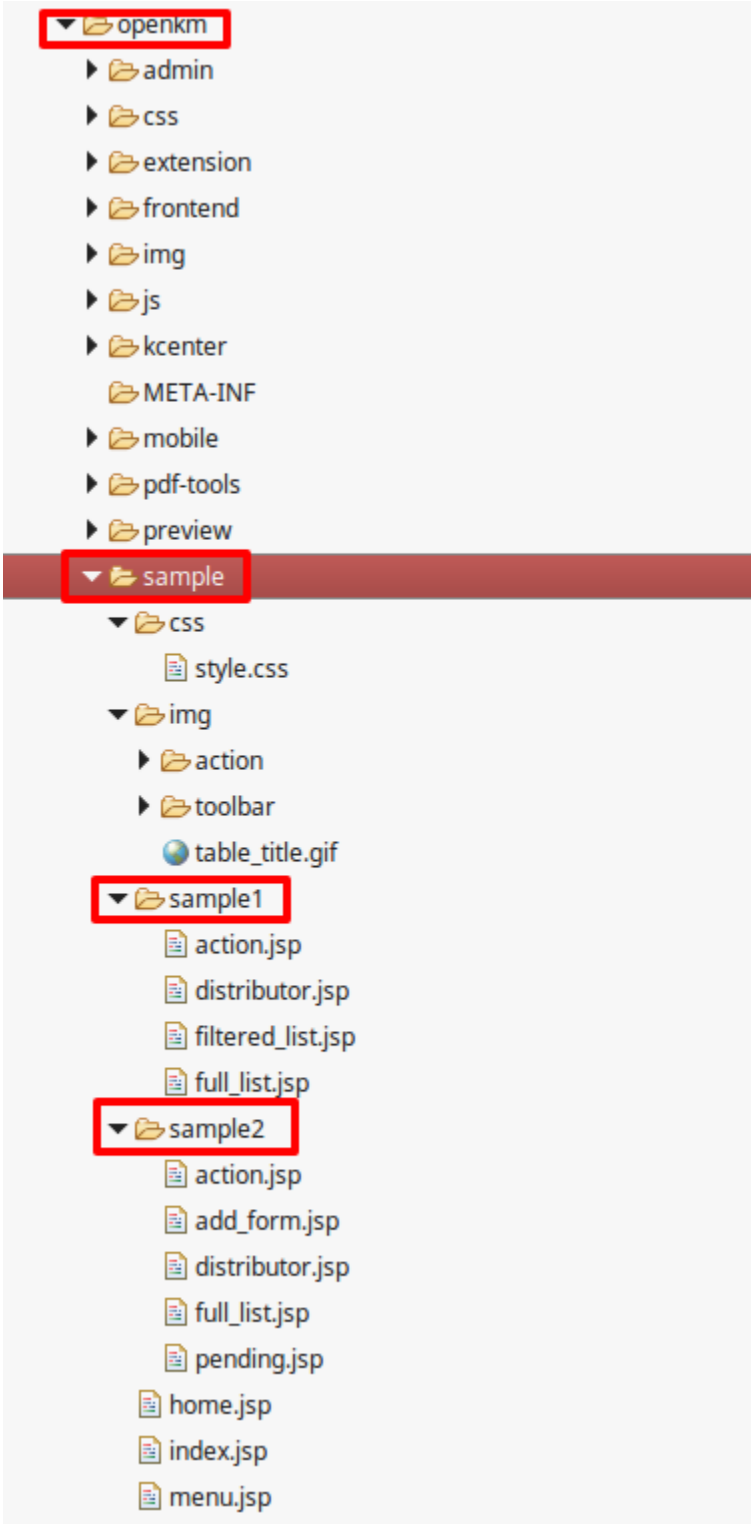
The sample is shown into a new Workspace tab.

The main screen is shown as a tool bar with tree buttons:

- *The first button goes to main screen of the sample.*
- *The second button show sample 1*
- *The second button show sample 2.*

Project structure:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Installation

- *Stop the application.*
- *Open the content of the download file into $TOMCAT_HOME/webapps/openkm folder (or better into openkm.war)*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Delete $TOMCAT_HOME/work/catalina/localhost*
- *Start the application.*
- *Check the url http://localhost:8080/openkm/sample.*

*The screen shots has been done in combination with "**ExtraTab**" frontend extension feature, also could be used TabWorkspaceExtension.*

When you change the contents of the openkm.war the file is deployed as a new version of the application and all the contents into $TOMCAT_HOME/webapps/openkm will be refreshed and you will lost any change you made directly in the webapps/openkm folder.

Implementation details

index.jsp

Index.jsp control the iframe height with a JavaScript code.

Other important fact is, for example after uploading a document (Sample 2), is used the parameter ***urlToOpen*** *to indicate which url must be opened.*

```
HttpSessionManager httpSessionManager = ContextWrapper.getContext().getBean(HttpSessi
OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
WebUtils webUtils = ContextWrapper.getContext().getBean(WebUtils.class);

httpSessionManager.add(request);
okmAuth.login();
String urlToOpen = URLDecoder.decode(webUtils.getString(request, "urlToOpen"), "UTF-8
if (urlToOpen.equals("")) {
    urlToOpen = "home.jsp";
}
```

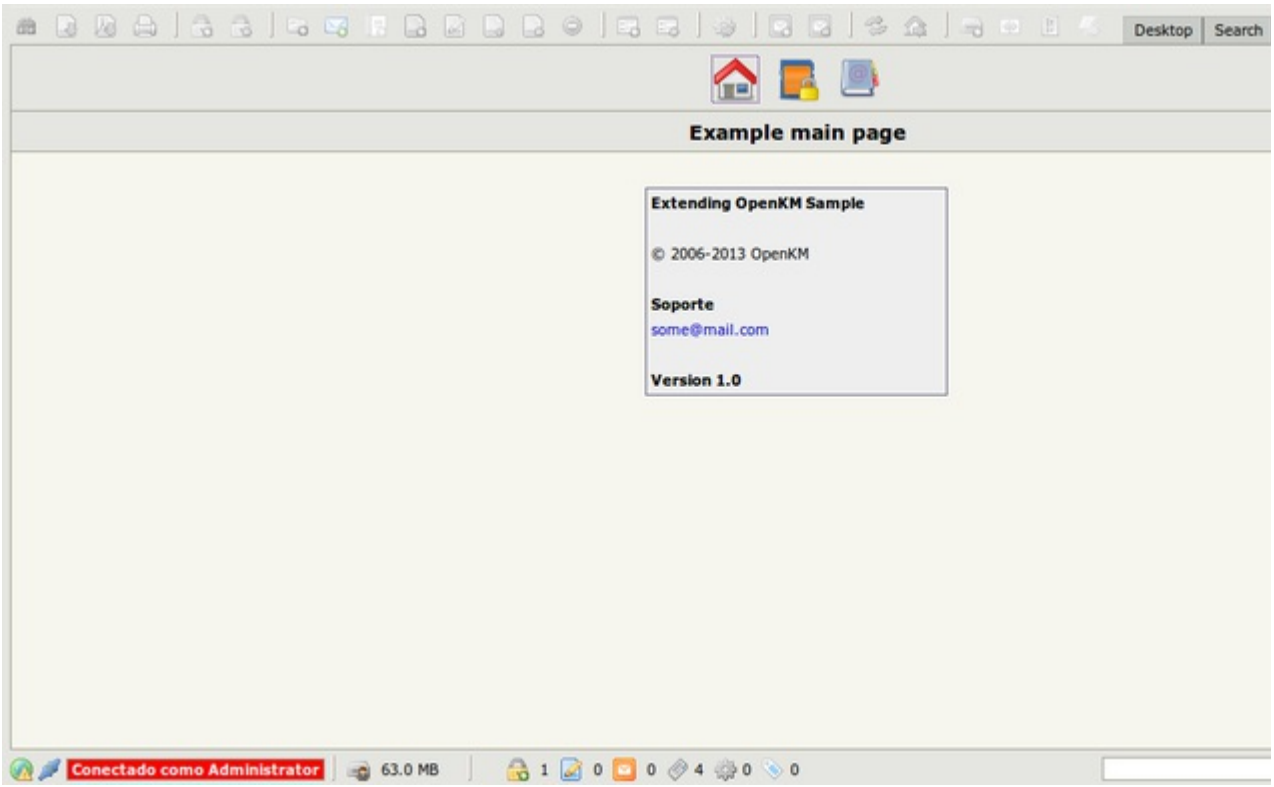
menu.jsp

Has the menu url definitions.

home.jsp

Set the main home screen.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Sample 1

Is composed by 4 files:

- *action.jsp (the main controller where is set the logic).*
- *distributor.jsp (the main layer with two columns at 50%).*
- *full_list.jsp (show full list with no JavaScript special control).*
- *filtered_list.jsp (show filtered list with no JavaScript special control).*

action.jsp

The main controller.

```
<%@page import="java.util.Map"%>
<%@page import="java.util.HashMap"%>
<%@page import="java.util.List"%>
<%@page import="java.util.ArrayList"%>
<%@ page import="java.text.SimpleDateFormat"%>
<%@ page import="java.io.IOException"%>
<%@ page import="java.net.URLDecoder"%>
<%@ page import="org.slf4j.Logger"%>
<%@ page import="org.slf4j.LoggerFactory"%>
<%@ page import="com.openkm.bean.Document"%>
<%@ page import="com.openkm.util.WebUtils"%>
<%@ page import="com.openkm.bean.CommonUser"%>
<%@ page import="com.openkm.api.OKMAuth"%>
<%@ page import="com.openkm.api.OKMRepository"%>
```

```
<%@ page import="com.openkm.api.OKMDocument"%>
<%@ page import="com.openkm.util.ContextWrapper"%>
<%@ page extends="com.openkm.extension.servlet.BaseServlet"%>
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
<%!
private static Logger log = LoggerFactory.getLogger("com.openkm.sample");
private static SimpleDateFormat df = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
private WebUtils webUtils = ContextWrapper.getContext().getBean(WebUtils.class);

public void doGet(HttpServletRequest request, HttpServletResponse response) throws IO
        ServletException {
    String action = webUtils.getString(request, "action");
    try {
        list(request, response);
    } catch (Exception e) {
        log.error(e.getMessage(), e);
        sendErrorRedirect(request, response, e);
    }
}

// Used by fileupload return
public void doPost(HttpServletRequest request, HttpServletResponse response) throws I
        ServletException {
    String action = webUtils.getString(request, "action");
    try {
        if (action.equals("filter")) {
            list(request, response);
        }
    } catch (Exception e) {
        log.error(e.getMessage(), e);
        sendErrorRedirect(request, response, e);
    }
}

private void list(HttpServletRequest request, HttpServletResponse response) throws Se
    OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

    String nameFilter = webUtils.getString(request, "nameFilter");
    String userFilter = webUtils.getString(request, "userFilter");

    // Getting lists
    List<Map<String, String>> leftDocsMaps = leftList(""); // Pending documents for a
    List<Map<String, String>> rightDocsMaps = rightList(userFilter, nameFilter);
    List<String> users = new ArrayList<String>();
    // UserList
    for (CommonUser user : okmAuth.getUsers(null)) {
        users.add(user.getId());
    }

    ServletContext sc = getServletContext();
    sc.setAttribute("nameFilter", nameFilter);
    sc.setAttribute("leftDocsMaps", leftDocsMaps);
    sc.setAttribute("rightDocsMaps", rightDocsMaps);
    sc.setAttribute("users", users);
    sc.setAttribute("nameFilter", nameFilter);
    sc.setAttribute("userFilter", userFilter);
    sc.getRequestDispatcher("/sample/sample1/distributor.jsp").forward(request, respo
}

private List<Map<String, String>> leftList(String user) throws Exception {
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    String uuid = okmRepository.getNodeUuid(null, "/okm:root");

    List<Map<String, String>> documentsMap = new ArrayList<Map<String, String>>();
    // To change
    List<Document> pendingDocuments = new ArrayList<Document>();
```

```
    for (Document doc : okmDocument.getChildren(null, uuid)) {
        Map<String, String> docMap = new HashMap<String, String>();
        docMap.put("author", doc.getAuthor());
        docMap.put("name", doc.getPath().substring(doc.getPath().lastIndexOf("/") + 1
        docMap.put("uuid", doc.getUuid());
        docMap.put("path", doc.getPath());
        docMap.put("mimeType", doc.getMimeType());
        docMap.put("lastModified", df.format(doc.getLastModified().getTime()));
        documentsMap.add(docMap);
    }
    return documentsMap;
}

private List<Map<String, String>> rightList(String user, String nameFilter) throws Ex
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    String uuid = okmRepository.getNodeUuid(null, "/okm:root");

    List<Map<String, String>> documentsMap = new ArrayList<Map<String, String>>();
    for (Document doc : okmDocument.getChildren(null, uuid)) {
        boolean found = true;
        if (!user.equals("") && !user.equals(doc.getAuthor())) {
            found = false;
        }
        if (!nameFilter.equals("") && !doc.getPath().contains(nameFilter)) {
            found = false;
        }

        if (found) {
            Map<String, String> docMap = new HashMap<String, String>();
            docMap.put("author", doc.getAuthor());
            docMap.put("name", doc.getPath().substring(doc.getPath().lastIndexOf("/")
            docMap.put("uuid", doc.getUuid());
            docMap.put("path", doc.getPath());
            docMap.put("mimeType", doc.getMimeType());
            docMap.put("lastModified", df.format(doc.getLastModified().getTime()));
            documentsMap.add(docMap);
        }
    }
    return documentsMap;
}
%>
```

distributor.jsp

The main layer.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="<%=request.getContextPath() %>/sample/
  <script type="text/javascript" src="<%=request.getContextPath() %>/js/jquery-1.7.1.
  <script type="text/javascript" src="<%=request.getContextPath() %>/js/vanadium-min.
</head>
<body>
 <table width="100%">
  <tbody>
  <tr>
   <td valign="top" width="50%">
    <jsp:include page="/sample/sample1/full_list.jsp"></jsp:include>
   </td>
```

```
    <td valign="top" width="10"></td>
    <td valign="top" width="50%">
     <jsp:include page="/sample/sample1/filtered_list.jsp"></jsp:include>
    </td>
   </tr>
   </tbody>
 </table>
</body>
</html>
```

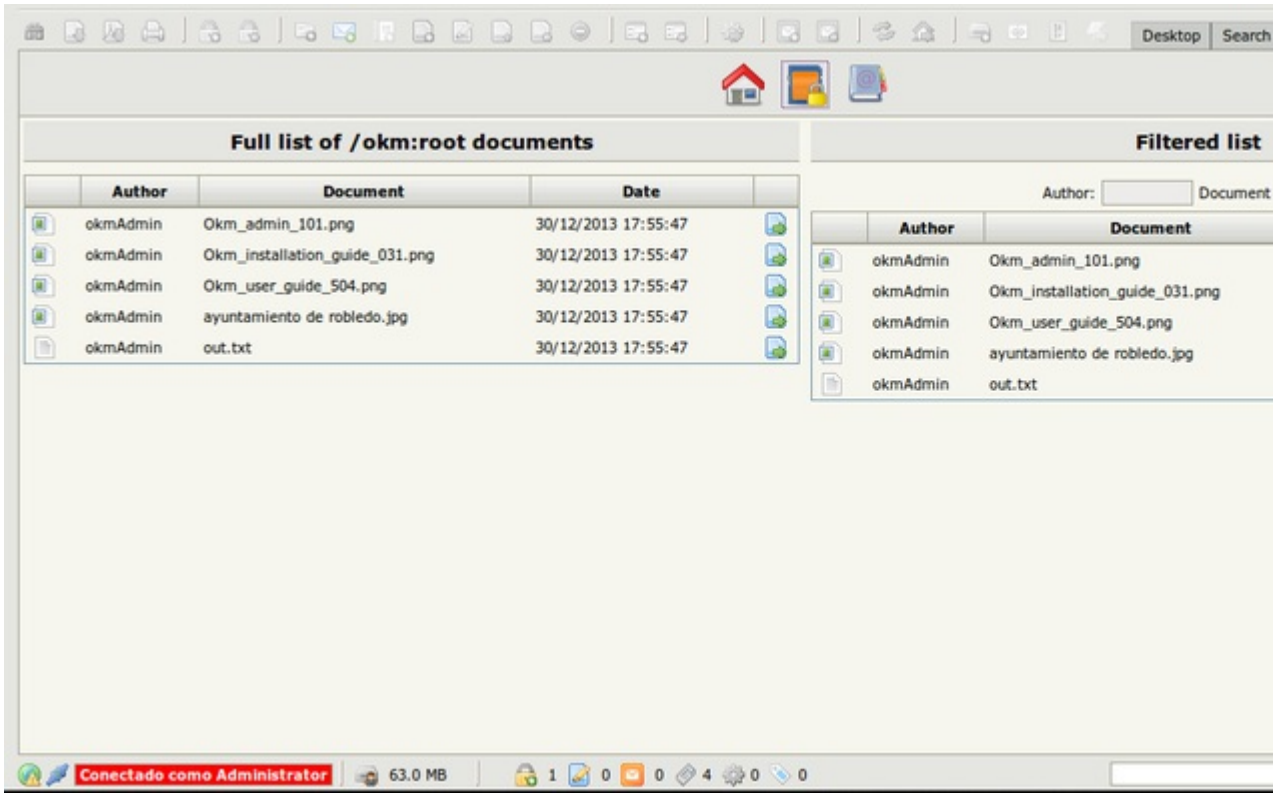
full_list.jsp

Show document list.

> *Pay attention at JavaScript function "**go**", which call parent.parent.jsOpenPathByUuid(uuid). The "**jsOpenPathByUuid**" is a GWT public function used to jump to some specific document, folder, etc. Here Exposed frontend JavaScript API there's a complete list of JavaScript functions.*

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<script type="text/javascript">
 // Note the parent.parent call because iframe is into other iframe
    function go(uuid) {
        alert('Document UUID:'+uuid);
     parent.parent.jsOpenPathByUuid(uuid);
    }
</script>
<h1>Full list of /okm:root documents</h1>
<table class="results" style="white-space: nowrap;" cellpadding="3" width="100%">
    <thead>
      <tr>
        <th></th><th>Author</th><th>Document</th><th>Date</th><th></th>
      </tr>
    </thead>
    <tbody>
     <c:forEach var="document" items="${leftDocsMaps}" varStatus="row">
     <tr>
      <td valign="center"><img src="<%=request.getContextPath() %>/mime/${document.mi
      <td>${document.author}</td>
      <td>${document.name}</td>
      <td>${document.lastModified}</td>
      <td align="center">
       <a href="javascript:go('${document.uuid}')"><img src="<%=request.getContextPat
       </td>
     </tr>
     </c:forEach>
    </tbody>
</table>
```

Screenshot of sample1 view



Sample 2

Is composed by 4 files:

- *action.jsp (the main controller where is set the logic).*
- *add_form.jsp (form to upload a new document)*
- *distributor.jsp (the main layer with two columns at 50% with JavaScript).*
- *full_list.jsp (show full list with JavaScript special control for table height).*
- *pending.jsp (static content).*

action.jsp

The main controller.

```
<%@page import="java.util.Map"%>
<%@page import="java.util.HashMap"%>
<%@page import="java.util.List"%>
<%@page import="java.util.ArrayList"%>
<%@ page import="java.text.SimpleDateFormat"%>
<%@ page import="java.io.IOException"%>
<%@ page import="org.slf4j.Logger"%>
<%@ page import="org.slf4j.LoggerFactory"%>
<%@ page import="com.openkm.bean.Document"%>
<%@ page import="com.openkm.util.WebUtils"%>
<%@ page import="com.openkm.api.OKMDocument"%>
```

```
<%@ page import="com.openkm.api.OKMRepository"%>
<%@ page import="com.openkm.util.ContextWrapper"%>
<%@ page extends="com.openkm.extension.servlet.BaseServlet"%>
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%
<%!
private static Logger log = LoggerFactory.getLogger("com.openkm.sample");
private WebUtils webUtils = ContextWrapper.getContext().getBean(WebUtils.class);

public void doGet(HttpServletRequest request, HttpServletResponse response) throws IO
    String action = webUtils.getString(request, "action");
    try {
        list(request, response);
    } catch (Exception e) {
        log.error(e.getMessage(), e);
        sendErrorRedirect(request, response, e);
    }
}

// Used by fileupload return and filter
public void doPost(HttpServletRequest request, HttpServletResponse response) throws I
    String action = webUtils.getString(request, "action");
    try {
        list(request, response);
    } catch (Exception e) {
        log.error(e.getMessage(), e);
        sendErrorRedirect(request, response, e);
    }
}

private void list(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException, Exception {
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    String uuid = okmRepository.getNodeUuid(null, "/okm:root");

    SimpleDateFormat df = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
    String docPath = (String) request.getAttribute("docPath");
    String nameFilter = webUtils.getString(request, "nameFilter");
    // Pending docs
    List<Map<String, String>> pendingDocMaps = new ArrayList<Map<String, String>>();
    for (Document doc : okmDocument.getChildren(null, uuid)) {
        Map<String, String> docMap = new HashMap<String, String>();
        docMap.put("name", doc.getPath().substring(doc.getPath().lastIndexOf("/") + 1
        docMap.put("uuid", doc.getUuid());
        docMap.put("path", doc.getPath());
        docMap.put("mimeType", doc.getMimeType());
        docMap.put("lastModified", df.format(doc.getLastModified().getTime()));
        pendingDocMaps.add(docMap);
    }
    ServletContext sc = getServletContext();
    sc.setAttribute("docPath", docPath);
    sc.setAttribute("nameFilter", nameFilter);
    sc.setAttribute("pendingDocuments", pendingDocMaps);
    sc.getRequestDispatcher("/sample/sample2/distributor.jsp").forward(request, respo
}
 %>
```

distributor.jsp

The main layer.

> *Here JavaScript is used to control the height and width of the layer.*

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="<%=request.getContextPath() %>/sample/
  <link rel="stylesheet" type="text/css" href="<%=request.getContextPath() %>/css/jqu
  <script type="text/javascript" src="<%=request.getContextPath() %>/js/jquery-1.7.1.
  <script type="text/javascript" src="<%=request.getContextPath() %>/js/vanadium-min.
  <script type="text/javascript" src="<%=request.getContextPath() %>/js/jquery.tables
</head>
<body>
 <div id="leftPanel" style="float: left;">
  <jsp:include page="/sample/sample2/add_form.jsp"></jsp:include>
  <jsp:include page="/sample/sample2/full_list.jsp"></jsp:include>
 </div>
 <div id="centerPanel" style="float: left; width: 12px;">
  <div id="verticalSeparator" class="verticalSeparator"></div>
 </div>
 <div id="rightPanel" style="float: left;">
        <jsp:include page="/sample/sample2/pending.jsp"></jsp:include>
 </div>

 <script type="text/javascript">
  var width =  $(window).width()-12; // Deleting middle panel
  var height = $(window).height();

  // Setting height
  $('#verticalSeparator').height(height);
  $('#leftPanel').height(height);
  $('#rightPanel').height(height);
  // Setting width
  $('#leftPanel').width(parseInt(width/2));
  $('#centerPanel').width(12);
  $('#rightPanel').width((width-parseInt((width/2))));
 </script>
</body>
</html>
```

add_form.jsp

Form to upload a new document.

```
<%@ page import="com.openkm.frontend.client.constants.ui.UIFileUploadConstants" %>
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<h2 id="headerTitle_new_register">Upload new file</h2>
<form enctype="multipart/form-data" method="post" action="<%=request.getContextPath()
 <input type="hidden" name="path" value="/okm:root">
 <input type="hidden" name="action" value="<%=UIFileUploadConstants.ACTION_INSERT%>">
 <input type="hidden" name="message" value="">
 <input type="hidden" name="comment" value="">
 <input type="hidden" name="redirect" value="/sample/sample2/action.jsp">
 <table class="form">
    <tbody>
     <c:if test="${docPath != null && docPath != ''}">
     <tr>
      <td><div class="ok">Document sended</div></td>
     </tr>
  </c:if>
     <tr>
      <td>
```

```
        <div id="file_error" style="display:none; color : red;">File is mandatory.</di
        <input type="file" name="file" class=":required;;file_error :only_on_blur">
       </td>
      </tr>
      <tr>
         <td align="center"><input type="submit" value="Send"></td>
      </tr>
    </tbody>
    </table>
</form>
```

full_list.jsp

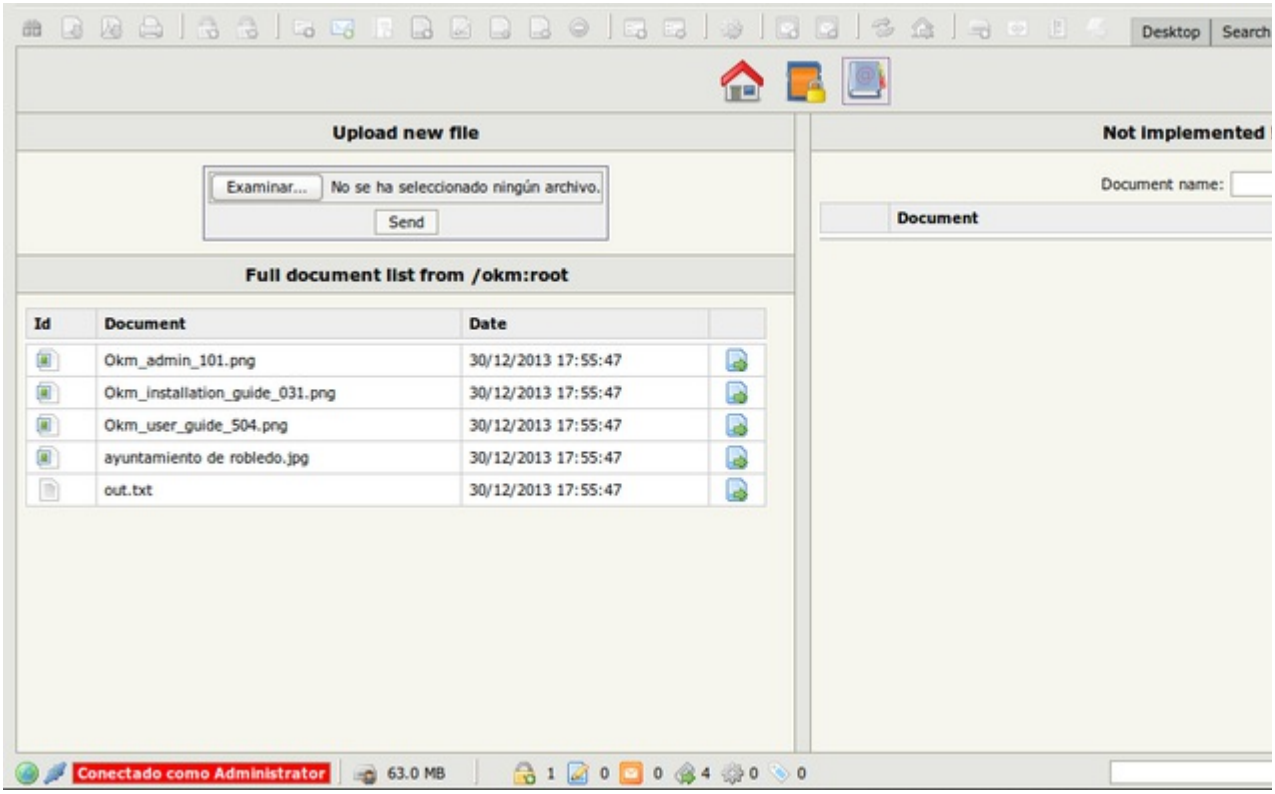
Show a list of documents.

> *Take a look at the ends where the JavaScript is used to control the scroll of the table.*

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<script type="text/javascript">
 // Note the parent.parent call because iframe is into other iframe
    function go(uuid) {
        alert('Document UUID:'+uuid);
     parent.parent.jsOpenPathByUuid(uuid);
    }
</script>
<h2 class="top">Full document list from /okm:root</h2>
<div align="center" style="padding: 0px 20px 0px 5px">
 <div class="tablescroll">
  <table id="docsPendingTable" width="100%" cellspacing="0" style="white-space: nowra
      <thead>
        <tr>
          <td>Id</td><td>Document</td><td>Date</td><td></td>
        </tr>
      </thead>
      <tbody>
       <c:set var="first" value="true"></c:set>
       <c:forEach var="document" items="${pendingDocuments}" varStatus="row">
        <c:choose>
          <c:when test="${first==true}">
           <tr class="first">
          </c:when>
          <c:otherwise>
           <tr>
          </c:otherwise>
        </c:choose>
         <td valign="center"><img src="<%=request.getContextPath() %>/mime/${document
         <td>${document.name}</td>
         <td>${document.lastModified}</td>
         <td align="center"><a href="javascript:go('${document.uuid}')"><img src="<%=
        </tr>
        <c:set var="first" value="false"></c:set>
       </c:forEach>
      </tbody>
  </table>
 </div>
</div>

<script type="text/javascript">
//Calculating table height removing from total other elements height
```

```
var pendingTableHeight = $(window).height() - (parseInt($('#docsPendingTable').offset
/*<![CDATA[*/
jQuery(document).ready(function($)
{
 $('#docsPendingTable').tableScroll({height:pendingTableHeight});
});
/*]]>*/
</script>
```

Screenshot of sample2 view



Using iframe for extending frontend UI

This sample gives some clues about how to use iframe in frontend UI. This sample use ExtraTab feature with an iframe that contains references to OpenKM object as links.

> *The HTML file has a valid OpenKM UUID's links.*
>
> *The sample below use Exposed frontend JavaScript API.*

File extra_tab.html:

```
<html>
<head>
  <title>Example</title>
</head>
<script type="text/javascript">
  function openPath(path) {
    parent.jsOpenPathByUuid(path);
  }
</script>
<body>
<h1>url</h1>
  <a href="javascript:void();" onclick="openPath('9789ac11-ac03-4ce7-87e9-9643b0ddf99
</body>
</html>
```

Copy the extra_tab.html file into $TOMCAT/webapps/OpenKM folder.

> *Copy to OpenKM folder is mandatory, otherside from iframe will not be able to access protected resource.*
>
> *When you change the contents of the OpenKM.war file is deployed a new version of the application and all the contents into $TOMCAT_HOME/webapps/OpenKM will be refreshed and you will lost any change you made directly in the webapps/OpenKM folder.*

Finally configure ExtraTab feature with the new url, for example http://localhost:8080/OpenKM/extra_tab.html.

> *Consider http://localhost:8080/OpenKM/extra_tab.html as an example. You should use your own public OpenKM application url.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Exposed frontend JavaScript API

OpenKM UI is developed with GWT framework what helps on calling Java methods from JavaScript.

The UI frontend Java methods are public exposed as JavaScript functions an can be used in frontend context.

More information about GWT JSNI at GWT Dev Codings Basics.

General functions

The methods that do not start with "js" will no longer be supported in the near future, we encourage you to move to js equivalent ones.

Function	*Return*	*Description*	*Deprecated*
jsI18n(String)	***void***	*Retrieve a frontend translation.* *The parameter is the translation key.*	*no*
jsOpenUserTaskInstance(String)	***void***	*Open dashboard workflow to view with user tas.* *The parameter is the taskinstance id.*	*no*
jsOpenPath(String, String)	***void***	*Open some node in OpenKM UI.* *First parameter is folder path.* *Second parameter is document path.* *When document path parameter is empty will be opened a folder.*	*no*
jsOpenPathByUuid(String)	***void***	*Open some node in OpenKM UI.* *The parameter is a valid node uuid (document, folder, mail or record).*	*no*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

jsRefreshFolder()	***void***	*Refresh UI (like toolbar refresh button action).*	*no*
jsWizard(String, String)	***void***	*Execute wizard.* *First parameter is a document path.* *Second parameter is a JSON to String serialization of GWTFileUploadResponse bean.* *The wizard usually should be called after a uploading a document.*	*no*
i18n(String)	***void***	*Retrieve a frontend translation.* *The parameter is the translation key.*	*yes*
openPath(String, String)	***void***	*Open some node in OpenKM UI.* *First parameter is folder path.* *Second parameter is document path.* *When document path parameter is empty will be opened a folder.*	*yes*
openPathByUuid(String)	***void***	*Open some node in OpenKM UI.* *The parameter is a valid node uuid (document, folder, mail or record).*	*yes*
refreshFolder()	***void***	*Refresh UI (like toolbar refresh button action).*	*yes*
jsCancelCheckout()	***void***	*Cancel document checkout of the selected document in UI.*	*no*
jsGetActualPath()	***String***	*Return the path of selected tree node in navigator.*	*no*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

jsGetActualNodePath()	***String***	*Return the path of the selected node in navigator or file brower.*	*no*
jsGetActualNodeUUID()	***String***	*Return the uuid of the selected node in navigator or file brower.*	*no*
jsGetUser()	***String***	*Return GWTUser object serialized to JSON.*	*no*
jsDownloadByUuid(String)	***void***	*Download a document.* *The parameter is the uuid of the document.*	*no*
jsChangeSelectedTab(int)	***void***	*Change workspace tab.* *Allowed values:* • *0 for Desktop.* • *1 for Seach.* • *2 for Dashboard.* • *3 for Administration.*	*no*

Cryptography functions

Function	*Return*	*Description*	*Deprecated*
cryptographyLoaded()	***void***	*Hide UI popup status that indicates cryptography applet is not yet loaded.*	*no*
digitalSignatureEnded()	***void***	*Indicate the digital signature has finished.*	*no*
digitalSignatureCanceled()	***void***	*Cancelling digital signature has been canceled.*	*no*
startDigitalSignature()	***void***	*Start digital signature.*	*no*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Applet functions

Function	Return	Description	Deprecated
destroyScannerApplet()	**void**	Cleaned the scanner applet.	no
destroyUploaderApplet()	**void**	Clean the bulk upload applet.	no

HTML Editor functions

Function	Return	Description	Deprecated
jsSearchDocumentHTMLEditorPopup()	**void**	Show find document popup.	no
jsSearchFolderHTMLEditorPopup()	**void**	Show find folder popup.	no

Wiki functions

Function	Return	Description	Deprecated
jsOpenWikiPage(String)	**void**	Open a wiki page The parameter is the unique wiki title.	no
openWikiPage(String)	**void**	Open a wiki page The parameter is the unique wiki title.	yes

How to expose Java methods to JavaScript

Static method sample

Expose **static method** named **i18n(String property)** to JavaScript as "**jsI18n(String)**" function:

```
/**
 * initJavaScriptApi
 */
native void initJavaScriptApi() /*-{
 $wnd.jsI18n = function(s) {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
   return @com.openkm.frontend.client.Main::i18n(Ljava/lang/String;)(s);
  };
 }-*/;
```

Non Static method sample

*Expose **non static method** "**executeRefresh()**" from class Toolbar.java to JavaScript as jsRefreshFolder function:*

*In the Toolbar Java create a method named "**initJavaScriptApi()**":*

```
public native void initJavaScriptApi(ToolBar toolBar) /*-{
 $wnd.jsRefreshFolder = function() {
  toolBar.@com.openkm.frontend.client.widget.toolbar.ToolBar::executeRefresh()();
  return true;
 }
}-*/;
```

*After creating a new Toolbar object call the "**initJavaScriptApi()**" method:*

```
ToolBar toolBar = new ToolBar();
mainPanel.topPanel.toolBar.initJavaScriptApi(mainPanel.topPanel.toolBar);
```

Complete information about GWT JSNI at GWT Dev Codings Basics.

Integration sample

*Create "**sample.jsp**" file:*

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
<html>
  <head>
    <title>Sample test</title>
    <script>
      function refresh() {
        alert('executing refresh');
        parent.parent.refreshFolder();
        alert('done');
      }
    </script>
  </head>
  <body>
    <a href="#" onclick="refresh()">Refresh UI</a>
  </body>
</html>
```

*Copy "**sample.jsp**" file into your "**$TOMCAT_HOME/webapps/openkm/**" folder.*

*Enable the extension named "**External Tab Document**", take a look at Enable extensions and External tab document extension.*

Remember you must enable the extension from the Profiles.

*Configure the parameter **extension.external.tab.document** with the value below.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Field / Property	*Type*	*Description*
extension.external.tab.document	***List***	*Each row is a couple of values with a tab content declaration, the format is:* ***name;url*** `Sample;http://localhost:8180/openkm/test.jsp`

More information at Configuration parameters.

Integrate preview with other web applications

You can integrate the OpenKM web preview module into other web applications quickly.

Advantatges:

- *Quickly integration into an iframe in your own web application.*
- *All supported OpenKM MIME types from preview modules.*
- *Can be used without being logged into OpenKM.*
- *Can preview documents which are not into OpenKM.*

*If your OpenKM is available from http://your_domain/openkm then, the preview OpenKM url module should be your **http://your_domain/openkm/Preview**?docUrl=http://localhost/test.pdf*

*The **supported MIME types depends on your OpenKM configuration and OpenKM conversion service** .*

This components previes most common file types:

- ***PDF, Microsoft Office** and **OpenOffice***
- ***Images like JPG, PNG, TIFF, MP4, MP3***
- ***Source code files like Java, PHP, HTML, SQL***

Preview parameters:

Parameter	*Type*	*Description*
docUrl	***String***	*The full url of a document.* *We recommend using it for all the file types.*
swfUrl	***String***	*The full url of a .swf file.*
dxfUrl	***String***	*The full url of a .dxf file.*
dwfUrl	***String***	*The full url of a .dwf file.*
htmUrl	***String***	*The full url of a .html file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

codUrl	***String***	*The full url of a file with code into, for example .java, .php, etc. files.*
mpgUrl	***String***	*The full url of a .mp4 and .mp3 file.*
pdfUrl	***String***	*The full url of a pdf document. Required by Flexpaper previewer.*
file	***String***	*The full url of a pdf document. Required by PDF.js previewer.* *PDF.js also require **pdfUrl** parameter*
previewEngine	***String***	*The type of preview to be used by server. By default is PDF.js but with this parameter you can force to Flexpaper* *OpenKM support Flexpaper and PDF.js as pdf previewers*

Code sample

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document Preview</title>
</head>

<body>
    <div align="left">
        Left panel
    </div>
    <div align="right">
        Preview panel<br />
        <iframe src="http://localhost:8080/openkm/Preview?docUrl=http://localhost/tes
            style="width:800px; height:600px" frameborder="0" marginwidth="0" marginh
    </div>
</body>

</html>
```

*If you **create your own downloading service** take in mind to set into the response the "Access-Control-Allow-Origin" in the header.*

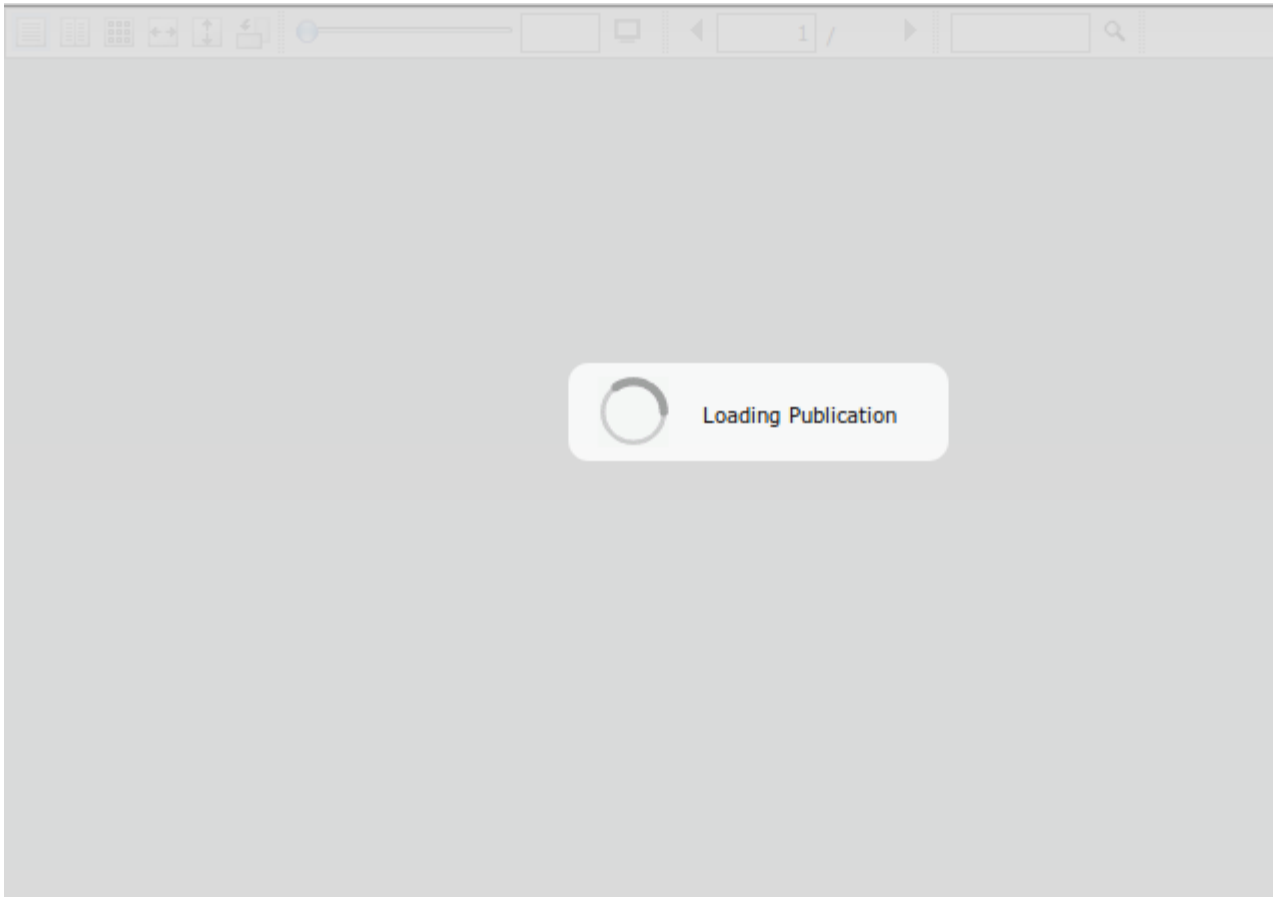
The java code section below can provide you some ideas about it:

```
/**
 * Prepare to send the file.
 */
public static void prepareSendFile(HttpServletRequest request, HttpServletRes
                    String mimeType, boolean inline) throws UnsupportedEncod
    String userAgent = getHeader(request, "user-agent").toLowerCase();
    fileName = decodeEntities(fileName);

    // Disable browser cache
    response.setHeader("Expires", "Sat, 6 May 1971 12:00:00 GMT");
    response.setHeader("Cache-Control", "must-revalidate");
    response.addHeader("Cache-Control", "no-store, no-cache, must-revalidate"
    response.addHeader("Cache-Control", "post-check=0, pre-check=0");
    response.setHeader("Pragma", "no-cache"); */

    // Allow to use files in different domains
    response.setHeader("Access-Control-Allow-Origin", "*");

    // Set MIME type
    response.setContentType(mimeType);

    if (userAgent.contains("msie") || userAgent.contains("trident")) {
        log.debug("Agent: Explorer ({})", request.getServerPort());
        fileName = URLEncoder.encode(fileName, "UTF-8").replaceAll("\\+", " "

        if (request.getServerPort() == 443) {
            log.debug("HTTPS detected! Apply IE workaround...");
            response.setHeader("Cache-Control", "max-age=1");
            response.setHeader("Pragma", "public");
        }
    } else if (userAgent.contains("iphone") || userAgent.contains("ipad")) {
        log.debug("Agent: iPhone - iPad");
        // Do nothing
    } else if (userAgent.contains("android")) {
        log.debug("Agent: Android");
        fileName = URLEncoder.encode(fileName, "UTF-8").replaceAll("\\+", " "
    } else if (userAgent.contains("mozilla")) {
        log.debug("Agent: Mozilla");
        fileName = MimeUtility.encodeText(fileName, "UTF-8", "B");
    } else {
        log.debug("Agent: Unknown");
    }

    if (inline) {
        response.setHeader("Content-disposition", "inline; filename=\"" + fil
    } else {
        response.setHeader("Content-disposition", "attachment; filename=\"" +
    }
}

/**
 * Get HTTP header.
 */
private static String getHeader(HttpServletRequest request, String name) {
    String value = request.getHeader(name);

    if (value != null) {
        return value;
    } else {
        return EMPTY_STRING;
    }
}

/**
 * decodeEntities
 */
```

```
private static String decodeEntities(String fileName) {
    String tmp = fileName.replaceAll("&amp;", "&");
    tmp = tmp.replaceAll("&lt;", "<");
    tmp = tmp.replaceAll("&gt;", ">");
    tmp = tmp.replaceAll("&#34;", "\"");
    tmp = tmp.replaceAll("&#39;", "'");
    return tmp;
}
```

Troubleshooting

There are several reasons why the document is not able for previewing and you get a screen like this:



Check if you see some error in your OpenKM log.

*Take a look in the javascript console, if it's shown an error like "**missing header CORS 'Access-Control-Allow-Origin')**" then you must change your web server configuration to allowing it.*

*At http://enable-cors.org/server_apache.html you have information about how to enable **Access-Control-Allow-Origin** in the Apache web server.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Samples

Sample	*Description*
Scripting samples	*A collection of samples that use Scripting feature.*
Crontab samples	*A collection of crontab samples.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Scripting samples

Script	*Description*
Script - Basic samples	• *Updates the repository statistics.* • *Creates folders and sets property group.* • *Shows the number of documents, folders and size from a given path.* • *Gets path by UUID.* • *Shows all documents used in every running workflow.*
Script - Compact documents with size equals to 0	*Finds all documents with size=0 and compact history of versions to latest.*
Script - Convert TIF to PDF	*Transform all the TIF files in the repository to PDF.*
Script - Count all children subfolder	*Counts all children folders - and sub folders to, recursion - into first level folder (okm:root children folders).*
Script - Exporter	*The repository exports execution.*
Script - Export repository folder to server as a ZIP file	*Exports the folder contents to the server application as a ZIP file.*
Script - Folders deep	*Finds the folders with some deep.*
Script - Import data from CSV file	*Imports metadata group values from csv file.*
Script - Purge all folders into some parent	*Purges all folders into some parent, without previously moving files to trash.*
Script - Purge all users trashes	*Example of how to clean all users trashes.*
Script - Read PDF form fields	*How to read PDF form fields.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Recursive character renaming	*Renames a character in all folders or documents from some initial path.*
Script - Recursive repository traversal	*Navigates across all repository nodes recursivelly.*
Script - Recursive change security	*Change the security.*
Script - Security change at low level	*Changes the security at database level.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Basic samples

Update repository statistics

Refresh the Statistics.

```
import com.openkm.core.RepositoryInfo;
import com.openkm.util.ContextWrapper;

RepositoryInfo repositoryInfo = ContextWrapper.getContext().getBean(RepositoryInfo.cl
repositoryInfo.run();
```

Create some folders and set property group

```
import java.util.HashMap;
import java.util.Map;

import com.openkm.api.OKMFolder;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(OKMPropertyGr

for (int i = 0; i < 10; i++) {
    String path = "/okm:root/fld_" + i;
    okmFolder.createSimple(null, path);

    Map<String, String> properties = new HashMap<>();
    properties.put("okp:consulting.name", "new name");
    properties.put("okp:consulting.comment", "new comment");
    okmPropertyGroup.addGroup(null, path, "okg:consulting", properties);
}
```

Show number of documents, folders and size from a given path

```
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.bean.ContentInfo;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FormatUtil;

OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
ContentInfo ci = okmFolder.getContentInfo(null, "151f3a54-f370-47d6-801a-d20faecec180

print("Folders: " + ci.getFolders() + "<br/>");
print("Documents: " + ci.getDocuments() + "<br/>");
print("Size: " + FormatUtil.formatSize(ci.getSize()) + "<br/>");
```

Get path by UUID

```
import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.class
String path = okmRepository.getNodePath(null, "162735a1-661c-4555-bf53-4e9115f38c62")
print(path);
```

Show all documents used in every running workflow

```
import com.openkm.api.OKMRepository;
import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;
import com.openkm.bean.workflow.ProcessInstance;
import com.openkm.core.Config;
import com.openkm.util.ContextWrapper;

OKMWorkflow okmWorkflow = ContextWrapper.getContext().getBean(OKMWorkflow.class);
for (ProcessDefinition procDef : okmWorkflow.findAllProcessDefinitions(null)) {
    for (ProcessInstance procIns : okmWorkflow.findProcessInstances(null, procDef.get
        if (procIns.getEnd() == null) {
            String uuid = (String) procIns.getVariables().get(Config.WORKFLOW_PROCESS_
            print("Doc UUID: " + uuid + ", Process Definition: " + procDef.getId() +
                    + procIns.getId());
        }
    }
}
```

Script - Compact documents with size equals to 0

Find all documents with size=0 and compact history of versions to latest.

```
import java.util.List;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import com.openkm.api.OKMDocument;
import com.openkm.api.OKMRepository;
import com.openkm.db.bean.NodeDocumentVersion;
import com.openkm.util.ContextWrapper;

SessionFactory sessionFactory = ContextWrapper.getContext().getBean(SessionFactory.cl
OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.class
OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);

String qs = "from NodeDocumentVersion ndv where ndv.size=0";
Session session = sessionFactory.openSession();
Transaction tx = session.beginTransaction();
Query q = session.createQuery(qs);
List docVersionList = q.list();
tx.commit();

print("Number of nodes: " + docVersionList.size() + "<br/>");

for (Object obj : docVersionList) {
    NodeDocumentVersion ndv = (NodeDocumentVersion) obj;
    okmDocument.purgeVersionHistory(null, ndv.getParent());
}

session.close();
print("done");
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Convert TIF to PDF

Transform all the TIF files in the repository to PDF.

- *There's an output log file named "ConvertTIFToPDFXXX.log" in the logs folders.*
- *Because large repositories the SQL query is executed with a database cursor.*

```
import org.springframework.web.context.*;
import com.openkm.module.db.base.*;
import com.openkm.util.*;
import com.openkm.servlet.frontend.util.ConversionUtils;
import org.hibernate.*;

WebApplicationContext cc = (WebApplicationContext) ContextWrapper.getContext();
SessionFactory sessionFactory = cc.getBean(SessionFactory.class);
ConversionUtils conversionUtils = cc.getBean(ConversionUtils.class);
StatelessSession session = sessionFactory.openStatelessSession();
String LOG_NAME = "ConvertTIFToPDF";
ScrollableResults results = null;

try {
  FileLogger.info(LOG_NAME, "### BEGIN conversion from TIF to PDF");
  String sql = "select nb.NBS_UUID from OKM_NODE_BASE nb where nb.NBS_NAME LIKE '%.ti
  Query query = session.createSQLQuery(sql);
  query.setReadOnly(true);
  query.setFetchSize(10);

  // Execute query
  results = query.scroll(ScrollMode.FORWARD_ONLY);

  while (results.next()) {
 String nodeUuid = results.get(0);
 try {
  String dstPath = conversionUtils.convertToPDF(nodeUuid);
  FileLogger.error(LOG_NAME, nodeUuid + " > " + dstPath);
 } catch (Exception e) {
  FileLogger.error(LOG_NAME, "Error in node " + nodeUuid + " caused by " + e.getMessa
  FileLogger.error(LOG_NAME + "-Error", "Error in node " + nodeUuid + " caused by \n"
 }
  }

  FileLogger.info(LOG_NAME, "### END conversion from TIF to PDF");
} catch (Exception e) {
  print(StackTraceUtils.toString(e));
  FileLogger.error(LOG_NAME, StackTraceUtils.toString(e));
} finally {
  results.close();
  session.close();
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Count all children subfolder

Count all children folders - and subfolders into, recursion - into first level folder (okm:root children folders).

```
import java.util.ArrayList;
import java.util.List;

import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import com.openkm.api.OKMFolder;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

class Util {

    public int childCount(String uuid) {
        String qsUUIDList = "select uuid from NodeFolder nf where nf.parent=:parent";
        Session session = null;
        Transaction tx = null;
        int value = 0;
        List uuidList = new ArrayList();

        try {
            SessionFactory sessionFactory = ContextWrapper.getContext().getBean(Sessi

            session = sessionFactory.openSession();
            tx = session.beginTransaction();
            Query q = session.createQuery(qsUUIDList).setCacheable(true);
            q.setString("parent", uuid);
            uuidList = q.list();
            tx.commit();
            value = uuidList.size();
            print(value);

        } catch (HibernateException e) {
            tx.rollback();
        } finally {
            session.close();
        }

        for (Object childUuid : uuidList) {
            value = value + childCount((String) childUuid);
        }
        return value;
    }
}


OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
Util util = new Util();
// /okm:root = 151f3a54-f370-47d6-801a-d20faecec180
for (Folder fld : okmFolder.getChildren(null, "151f3a54-f370-47d6-801a-d20faecec180")
    print(fld.getPath() + "->" + util.childCount(fld.getUuid()) + "<br/>");
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Exporter

Repository export execution.

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

import com.openkm.api.OKMFolder;
import com.openkm.bean.ContentInfo;
import com.openkm.core.Config;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileLogger;
import com.openkm.util.impexp.HTMLInfoDecorator;
import com.openkm.util.impexp.RepositoryExporter;

OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
RepositoryExporter repositoryExporter = ContextWrapper.getContext().getBean(Repositor

// Configuration parameters
String token = DbSessionManager.getInstance().getSystemToken();
String repoPath = "/okm:root";
String fsPath = "/home/openkm/export";
boolean history = false;

// Start
FileWriter fw = null;
try {
    FileLogger.info("LOG_FILE_NAME", "Started");
    // Open outputstream
    String fileDate = new SimpleDateFormat("yyyyMMdd").format(new Date());
    String fileName = Config.LOG_DIR + File.separator + "LOG_FILE_NAME" + "_" + fileD
    File file = new File(fileName);
    fw = new FileWriter(file);

    // Starting the process
    ContentInfo cInfo = okmFolder.getContentInfo(token, repoPath);
    FileLogger.info("LOG_FILE_NAME",
            "Files & directories to export:" + (cInfo.getDocuments() + cInfo.getFolde
    File dir = new File(fsPath);
    repositoryExporter.exportFolder(token, repoPath, dir, history, fw,
            new HTMLInfoDecorator((int) cInfo.getDocuments() + (int) cInfo.getFolders
    FileLogger.info("CrontabRepositoryExporter", "Finished");
} catch (Exception e) {
    try {
        if (fw != null) {
            fw.close();
        }
    } catch (IOException e1) { // Ignore
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Export repository folder to server as a ZIP file

This script exports all contents of an OpenKM folder to application server as a ZIP file.

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.StringWriter;

import org.apache.commons.io.IOUtils;

import com.openkm.util.ArchiveUtils;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.impexp.RepositoryExporter;
import com.openkm.util.impexp.TextInfoDecorator;

FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class);
RepositoryExporter repositoryExporter = ContextWrapper.getContext().getBean(Repositor
ArchiveUtils archiveUtils = ContextWrapper.getContext().getBean(ArchiveUtils.class);

// Destination
File file = new File("/home/openkm/import/folderExport.zip");
// OpenKM folder contents to be exported
String fldPath = "/okm:root/invoices";

FileOutputStream os = new FileOutputStream(file);
StringWriter out = new StringWriter();
File tmp = fileUtils.createTempDir();
repositoryExporter.exportFolder(null, fldPath, tmp, false, out, new TextInfoDecorator
archiveUtils.createZip(tmp, "import", os);

org.apache.commons.io.FileUtils.deleteDirectory(tmp);
IOUtils.closeQuietly(out);
os.flush();
os.close();
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Folders deep

The script finds all folders with some deep.

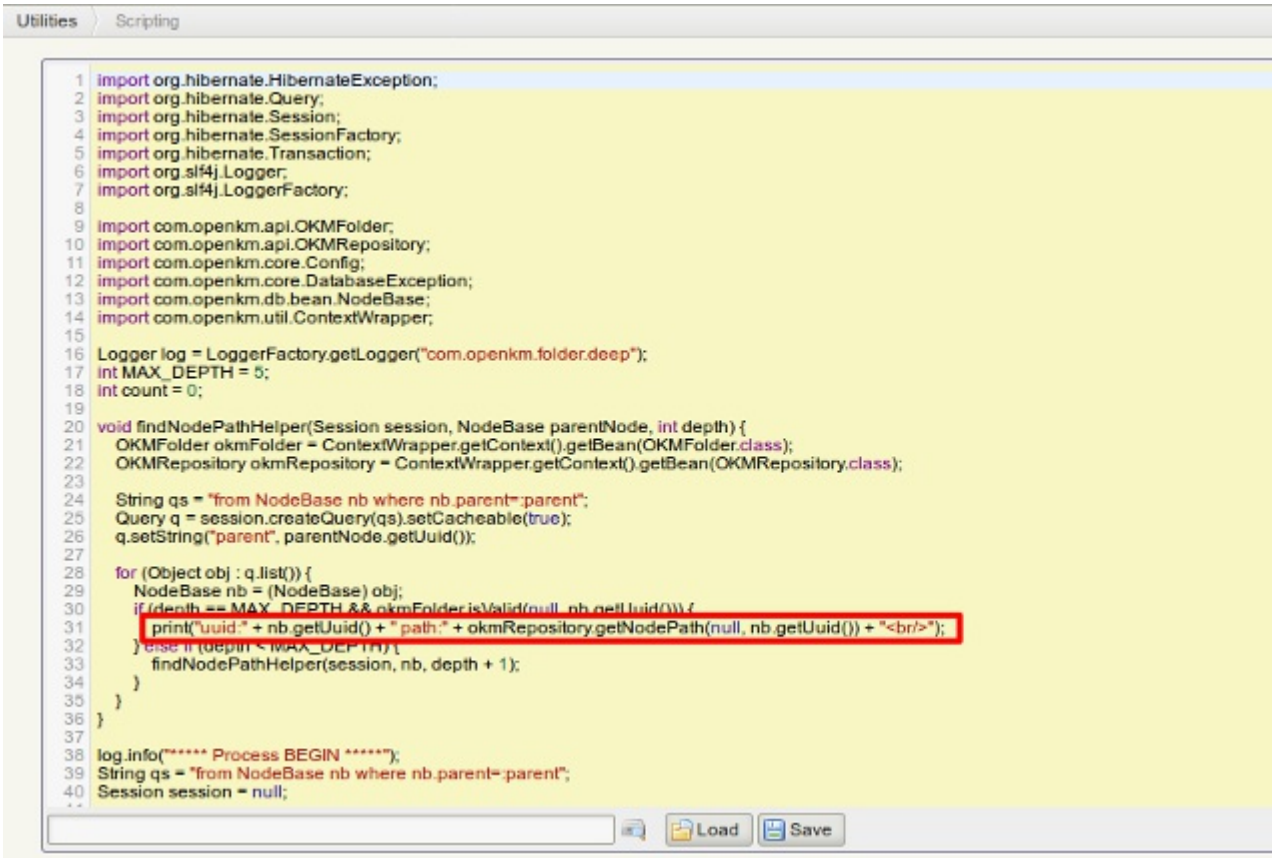
> *The variable **MAX_DEPTH=5** indicates the folder deep.*

```
import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.openkm.api.OKMFolder;
import com.openkm.api.OKMRepository;
import com.openkm.core.Config;
import com.openkm.core.DatabaseException;
import com.openkm.db.bean.NodeBase;
import com.openkm.util.ContextWrapper;

Logger log = LoggerFactory.getLogger("com.openkm.folder.deep");
int MAX_DEPTH = 5;
int count = 0;

void findNodePathHelper(Session session, NodeBase parentNode, int depth) {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c

    String qs = "from NodeBase nb where nb.parent=:parent";
    Query q = session.createQuery(qs).setCacheable(true);
    q.setString("parent", parentNode.getUuid());

    for (Object obj : q.list()) {
        NodeBase nb = (NodeBase) obj;
        if (depth == MAX_DEPTH && okmFolder.isValid(null, nb.getUuid())) {
            print("uuid:" + nb.getUuid() + " path:" + okmRepository.getNodePath(null,
        } else if (depth < MAX_DEPTH) {
            findNodePathHelper(session, nb, depth + 1);
        }
    }
}

log.info("***** Process BEGIN *****");
String qs = "from NodeBase nb where nb.parent=:parent";
Session session = null;

log.info("***** Process BEGIN *****");
String qs = "from NodeBase nb where nb.parent=:parent";
Session session = null;
Transaction tx = null;

try {
    SessionFactory sessionFactory = ContextWrapper.getContext().getBean(SessionFactor

    session = sessionFactory.openSession();
    tx = session.beginTransaction();

    // First level nodes
    Query q = session.createQuery(qs);
```

```
    q.setString("parent", Config.ROOT_NODE_UUID);

    for (Object obj : q.list()) {
        NodeBase nb = (NodeBase) obj;
        // Process in depth
        findNodePathHelper(session, nb, 0);
    }
    tx.commit();
} catch (HibernateException e) {
    tx.rollback();
    throw new DatabaseException(e.getMessage(), e);
} finally {
    session.close();
}

log.info("***** Process END *****");
```

Images

Execution



Results

Script error
Script result
Script output
uuid:e18393ad-912f-4b0a-8095-596456d52ae8 path:/okm:root/test 1.2/test2/test3/test4/test5 uuid:c0b84029-da90-472a-80e5-44819b2a47f3 path:/okm:root/test 1.1/test2/test3/test4/test5 uuid:5dbc701b-658e-4e4b-ba8b-710a7e831fac path:/okm:root/test 1.3/test2/test3/test4/test5 uuid:70606548-586e-4a10-94c1-7b32d7734f4d path:/okm:trash/okmAdmin/test 1.4/test2/test3/test4

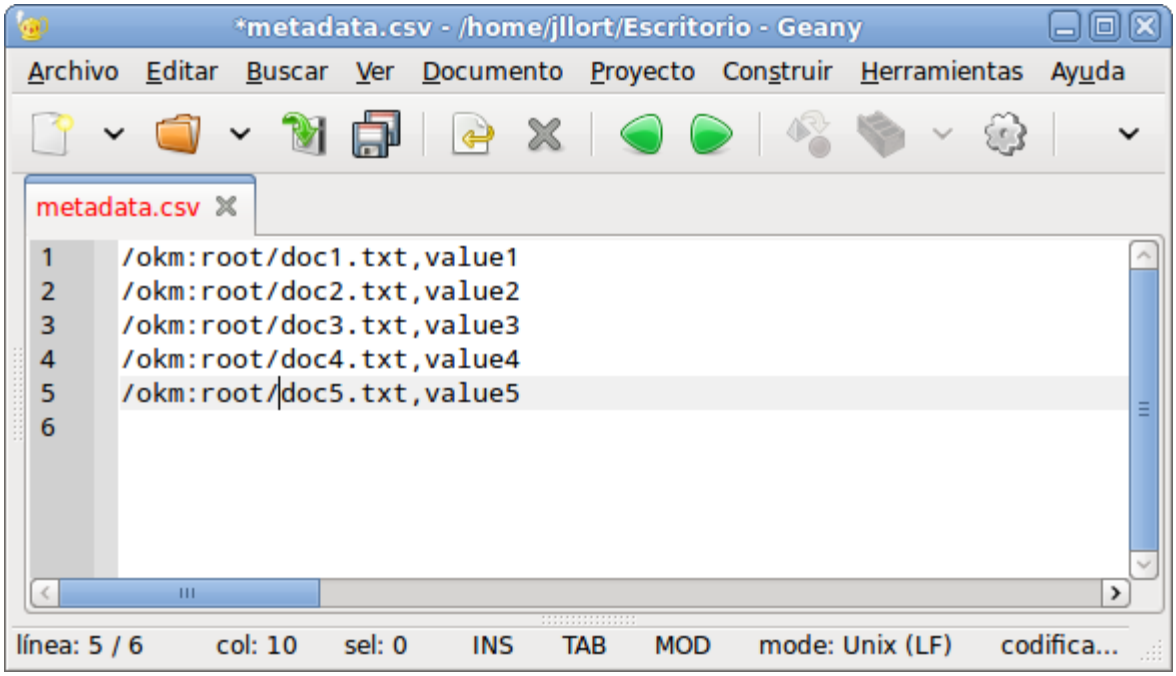
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Import data from CSV file

The script can be executed from the administration scripting (or used as automation script too) to import metadata values.

- *There's a metadata CSV file with two columns, one for the document path and other for the metadatada value. The script can be easily changed to use more than two columns.*
- *Script searches documents by document name. In theory there's only one document with the same name in the system, otherwise script shows error. Here could be used document path from first csv column, but to get more complete sample it has been used a search engine to find document.*
- *For each document found, script add metadata group with CSV values.*

The CSV file has two colum, first column contains document path and second column has metadata value.



Download CSV sample file csv_metadata.zip.

The metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.1//EN"
"http://www.openkm.com/dtd/property-groups-2.1.dtd">
<property-groups>
    <property-group label="Metadata" name="okg:metadata">
        <input label="Document ID" name="okp:metadata.value" type="text"/>
    </property-group>
</property-groups>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The script:

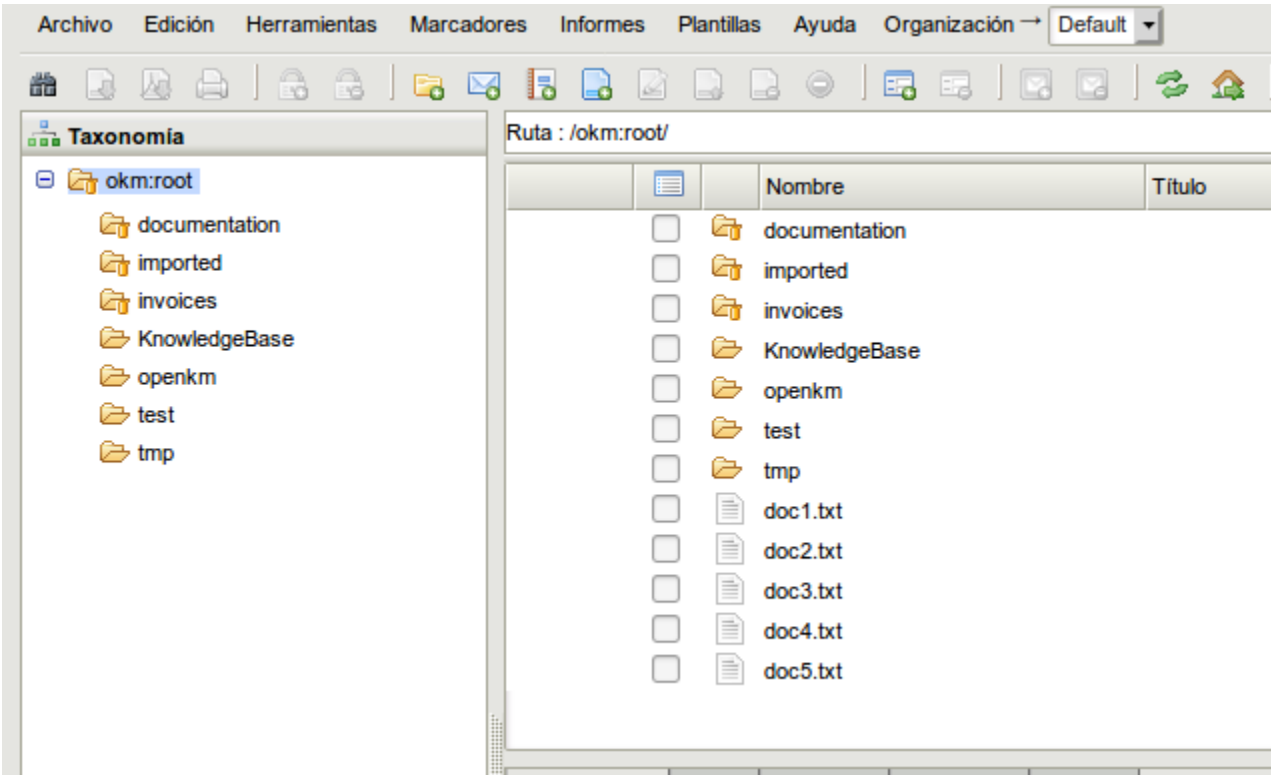
```
import java.io.FileReader;
import java.io.Reader;
import java.util.Collection;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;

import com.googlecode.jcsv.CSVStrategy;
import com.googlecode.jcsv.reader.CSVReader;
import com.googlecode.jcsv.reader.internal.CSVReaderBuilder;
import com.googlecode.jcsv.reader.internal.DefaultCSVEntryParser;
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.api.OKMSearch;
import com.openkm.bean.Document;
import com.openkm.bean.QueryResult;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;

String grpName = "okg:metadata";
String FILE_LOG_NAME = "CSVLOG";
String META_PATH = "/home/openkm/csv/";
String META_FILE_NAME = "metadata.csv";
int uniqueFileName = 0;
int valueColumn = 1;

// Format defintion
char delimiter = ',';
char quoteCharacter = '"';
char commentIndicator = '#';
boolean skipHeader = false;
boolean ignoreEmptyLines = true;
CSVStrategy strategy = new CSVStrategy(delimiter, quoteCharacter, commentIndicator, s
// File reader
Reader reader = new FileReader(META_PATH + META_FILE_NAME);
// CSV reader
CSVReader csvParser = new CSVReaderBuilder(reader).strategy(strategy).entryParser(new
List data = csvParser.readAll();
int count = 0;
int countFound = 0;
int countNotDocument = 0;
int moreThanOneDocumentFound = 0;
int notFound = 0;
int noName = 0;

OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class);
OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(OKMPropertyGr
PathUtils pathUtils = ContextWrapper.getContext().getBean(PathUtils.class);

for (int i = 0; i < data.size(); i++) {
    String[] row = (String[]) data.get(i);
    String docPath = row[uniqueFileName];
    print(count + ">>>> " + docPath);

    if (docPath != null && !docPath.equals("")) {
        QueryParams queryParams = new QueryParams();
        queryParams.setDomain(QueryParams.DOCUMENT);
        queryParams.setName(pathUtils.getName(docPath));
        Collection results = okmSearch.find(null, queryParams);

        if (results.size() == 1) {
```

```
            QueryResult queryResult = (QueryResult) results.iterator().next();
            if (queryResult.getNode() != null && queryResult.getNode() instanceof Doc
                print("found");
                countFound++;

                Map properties = new HashMap();
                properties.put("okp:metadata.value", row[valueColumn]);

                // Add Group and metadata
                okmPropertyGroup.addGroup(null, docPath, grpName, properties);
            } else {
                print("error is not document");
                countNotDocument++;
            }
        } else if (results.size() > 1) {
            print("error more than one document found can not decide");
            moreThanOneDocumentFound++;
        } else {
            print("not found");
            notFound++;
        }
    } else {
        print("error document has no name");
        noName++;
    }

    print("</br>");

    count++;
}

print("Total:" + count + "</br>");
print("Found:" + countFound + "</br>");
print("Error not document:" + countNotDocument + "</br>");
print("Error more then one document found:" + moreThanOneDocumentFound + "</br>");
print("Error not found:" + notFound + "</br>");
print("Error name empty:" + notFound + "</br>");
```

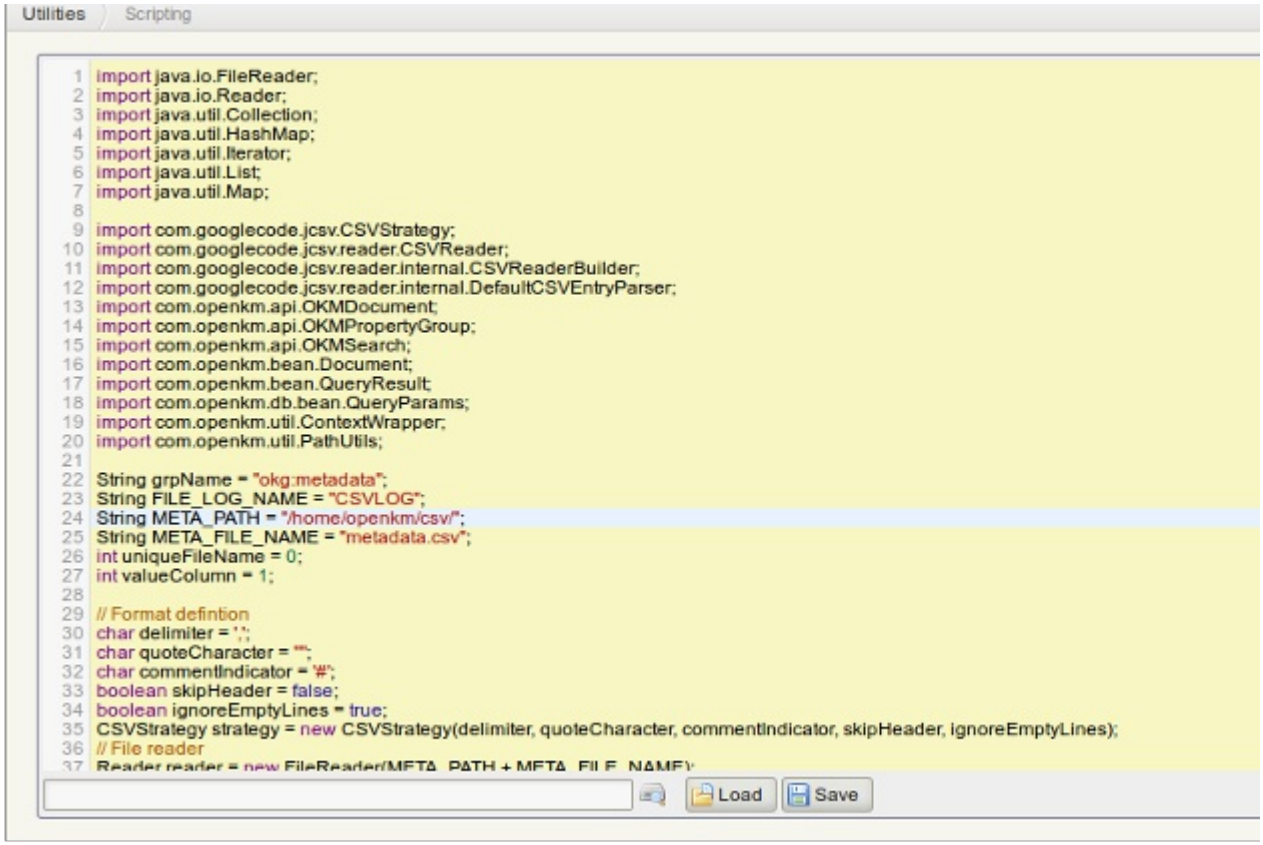
Image

Register the metadata group

Repository view



Execution

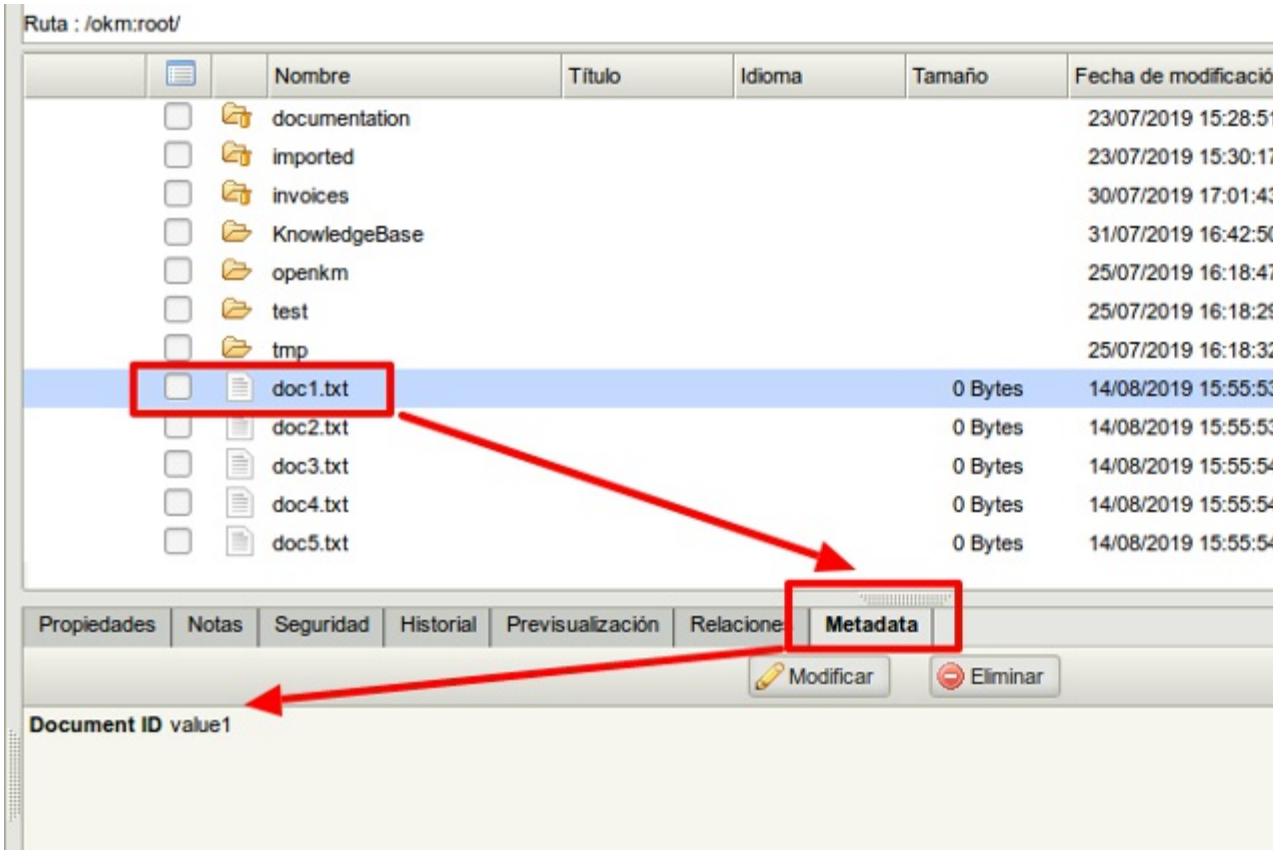


Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script results

Script error
Script result
Script output
0>>>> /okm:root/doc1.txt found 1>>>> /okm:root/doc2.txt found 2>>>> /okm:root/doc3.txt found 3>>>> /okm:root/doc4.txt found 4>>>> /okm:root/doc5.txt found Total:5 Found:5 Error not document:0 Error more then one document found:0 Error not found:0 Error name empty:0

Imported metadata



Script - Purge all folders into some parent

That script purges all folders into a parent, without previously moving files to trash.

The purge action is a non recovery action that removes definitely the objects from OpenKM repository. Use it with care.

```
import java.util.ArrayList;
import java.util.List;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import com.openkm.api.OKMFolder;
import com.openkm.db.bean.NodeFolder;
import com.openkm.util.ContextWrapper;

String qs = "from NodeFolder nf where nf.parent='2ed9586e-570c-4d27-a107-eeb0a5f181be
Session session = null;
Transaction tx = null;
boolean found = true;

try {
    SessionFactory sessionFactory = ContextWrapper.getContext().getBean(SessionFactor
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);

    while (found) {

        session = sessionFactory.openSession();
        Query q = session.createQuery(qs);
        q.setMaxResults(100);
        found = false;
        List fldList = new ArrayList();

        for (NodeFolder fld : q.list()) {
            found = true;
            fldList.add(fld.getUuid());
        }

        for (String uuid : fldList) {
            try {
                okmFolder.purge(null, uuid);
            } catch (Exception e) {
                print(e.getMessage() + "<br/>");
            }
        }
    }
} catch (Exception e) {
    print(e.getMessage());
} finally {
    session.close();
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Purge all users trashes

```
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMMail;
import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.util.ContextWrapper;

try {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);

    String token = DbSessionManager.getInstance().getSystemToken();

    for (Folder trash : okmFolder.getChildren(token, "/okm:trash")) {
        print("Trash: " + trash.getPath() + "<br/>");

        for (Folder fld : okmFolder.getChildren(token, trash.getPath())) {
            print("About to delete folder: " + fld.getPath() + "<br/>");
            okmFolder.purge(token, fld.getUuid());
        }

        for (Document doc : okmDocument.getChildren(token, trash.getPath())) {
            print("About to delete document: " + doc.getPath() + "<br/>");

            if (okmDocument.isLocked(token, doc.getUuid())) {
                okmDocument.forceUnlock(token, doc.getUuid());
            }

            okmDocument.purge(token, doc.getUuid());
        }

        for (Mail mail : okmMail.getChildren(token, trash.getUuid())) {
            print("About to delete mail: " + mail.getPath() + "<br/>");
            okmMail.purge(token, mail.getUuid());
        }
    }
} catch (Exception e) {
    print(e.getMessage());
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Put documents from folder and descendants into pending text extractio

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.openkm.api.OKMFolder;
import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.db.service.LegacySrv;
import com.openkm.util.ContextWrapper;

Logger log = LoggerFactory.getLogger("com.openkm.scripting");
int MAX_DEPTH = Integer.MAX_VALUE;

void nodeTask(String uuid, int depth) throws Exception {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    LegacySrv legacySrv = ContextWrapper.getContext().getBean(LegacySrv.class);

    String sql = "UPDATE OKM_NODE_DOCUMENT SET NDC_TEXT_EXTRACTED='F' WHERE NBS_UUID
            + uuid + "')";
    legacySrv.executeSQL(sql);

    for (Folder fld : okmFolder.getChildren(null, uuid)) {
        log.info("Folder: {}", fld.getPath());

        if (depth < MAX_DEPTH) {
            nodeTask(fld.getUuid(), depth + 1);
        }
    }
}

try {
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    log.info("***** Process BEGIN *****");
    String fldPath = "/okm:root/KnowledgeBase";
    String fldUuid = okmRepository.getNodeUuid(null, fldPath);
    nodeTask(fldUuid, 0);
    log.info("***** Process END *****");
} catch (Exception e) {
    print(e.getMessage());
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Read PDF form fields

This is a script what demonstrates how to read PDF form fields. Sample document formtempl.pdf.

```
import com.lowagie.text.pdf.*;
import com.openkm.util.*;
import com.openkm.api.*;
import org.apache.commons.io.IOUtils;
import java.io.InputStream;

OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
InputStream is = null;
try {
 is = okmDocument.getContent(null, "e5a84c26-f8ed-4e7a-8a05-257ea4db3569", false); //
 PdfReader reader = new PdfReader(is);
 PRAcroForm form = reader.getAcroForm();
 AcroFields fields = reader.getAcroFields();

 if (form != null) {
  for (PRAcroForm.FieldInformation field : form.getFields()) {
   String fieldName = field.getName();
   String fieldValue = fields.getField(fieldName);
   print("Field: " + fieldName + ", Value: " + fieldValue);
  }
 }
} finally {
 IOUtils.closeQuietly(is);
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Recursive character renaming

Renames a character in all folders or documents from some initial path.

```
// Recursive renaming & to -
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMRepository;
import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;

void findNodePathHelper(String uuid) throws Exception {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    PathUtils pathUtils = ContextWrapper.getContext().getBean(PathUtils.class);

    for (Folder fld : okmFolder.getChildren(null, uuid)) {
        String fldName = pathUtils.getName(fld.getPath());

        if (fldName.contains("&")) {
            print(fld.getPath() + "<br>");
            String newName = fldName.replaceAll("&", "-");
            okmFolder.rename(null, fld.getUuid(), newName);
        }

        for (Document doc : okmDocument.getChildren(null, fld.getPath())) {
            String docName = pathUtils.getName(doc.getPath());

            if (docName.contains("&")) {
                print(doc.getPath() + "<br>");
                String newName = docName.replaceAll("&", "-");
                okmDocument.rename(null, doc.getUuid(), newName);
            }
        }

        findNodePathHelper(fld.getUuid());
    }
}

try {
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    String fldPath = "/okm:root";
    String fldUuid = okmRepository.getNodeUuid(null, fldPath);
    findNodePathHelper(fldUuid);
} catch (Exception e) {
    print(e.getMessage());
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Recursive repository traversal

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMRepository;
import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

Logger log = LoggerFactory.getLogger("com.openkm.scripting");
int MAX_DEPTH = Integer.MAX_VALUE;

void nodeTask(String uuid, int depth) throws Exception {
    OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);

    for (Document doc : okmDocument.getChildren(null, uuid)) {
        log.info("Document: {}", doc.getPath());
        print("Document: " + doc.getPath() + "<br>");
    }

    for (Folder fld : okmFolder.getChildren(null, uuid)) {
        log.info("Folder: {}", fld.getPath());
        print("Document: " + fld.getPath() + "<br>");

        if (depth < MAX_DEPTH) {
            nodeTask(fld.getUuid(), depth + 1);
        }
    }
}

log.info("***** Process BEGIN *****");
try {
    OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepository.c
    String fldPath = "/okm:root";
    String fldUuid = okmRepository.getNodeUuid(null, fldPath);
    nodeTask(fldUuid, 0);
} catch (Exception e) {
    print(e.getMessage());
}
log.info("***** Process END *****");
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script - Recursive change security

```
import java.util.HashMap;
import java.util.Map;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.openkm.api.OKMAuth;
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMMail;
import com.openkm.api.OKMRecord;
import com.openkm.bean.CommonUser;
import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.bean.Permission;
import com.openkm.bean.Record;
import com.openkm.core.AccessDeniedException;
import com.openkm.core.DatabaseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.util.ContextWrapper;

Logger log = LoggerFactory.getLogger("com.openkm.scripting");
int MAX_DEPTH = Integer.MAX_VALUE;
Map grantUsers = new HashMap();
Map grantRoles = new HashMap();
Map revokeUsers = new HashMap();
Map revokeRoles = new HashMap();

OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class);
OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class);
OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

void nodeTask(String uuid, int depth) throws Exception {
    setSecurity(uuid);
    for (Document doc : okmDocument.getChildren(null, uuid)) {
        setSecurity(doc.getUuid());
        log.info("Document: {}", doc.getPath());
    }

    for (Folder fld : okmFolder.getChildren(null, uuid)) {
        if (depth < MAX_DEPTH) {
            nodeTask(fld.getUuid(), depth + 1);
        }
    }

    for (Mail mail : okmMail.getChildren(null, uuid)) {
        setSecurity(mail.getUuid());
        log.info("Mail: {}", mail.getPath());
    }

    for (Record rec : okmRecord.getChildren(null, uuid)) {
        log.info("Record: {}", rec.getPath());

        if (depth < MAX_DEPTH) {
            nodeTask(rec.getUuid(), depth + 1);
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
void setSecurity(String uuid) throws PathNotFoundException, AccessDeniedException, Da
    // remove all
    okmAuth.changeSecurity(null, uuid, new HashMap(), revokeUsers, new HashMap(), rev
    // set specific
    okmAuth.changeSecurity(null, uuid, grantUsers, new HashMap(), grantRoles, new Has
}

try {
    log.info("***** Process BEGIN *****");
    String parentUuid = "162735a1-661c-4555-bf53-4e9115f38c62"; // Choose your folder
    // Loading users and roles to be removed
    for (CommonUser user : okmAuth.getUsers(null)) {
        revokeUsers.put(user.getId(), Permission.ALL);
    }
    for (String role : okmAuth.getRoles(null)) {
        revokeRoles.put(role, Permission.ALL);
    }

    // Loading users and roles to be added ( allowed permissions are ALL = READ + WRI
    grantRoles.put("ROLE_USER", Permission.READ);
    grantUsers.put("okmAdmin", Permission.ALL);
    grantUsers.put("sochoa", Permission.READ + Permission.WRITE);

    nodeTask(parentUuid, 0);
    log.info("***** Process END *****");
} catch (Exception e) {
    print(e.getMessage());
}
```

Script - Security change at low level

This is an administrator script that should be used only when to propagate security changes under a folder which have a significant number of nodes is needed. These type of changes made from UI can take a lot of time to be propagated across the repository and this script does it much faster.

*Our recommendation is to use it when changes affects more than **100.000 nodes** otherwise we recommend do it from UI.*

The security change starts with some folder UUID node and propagates across all children. The logic is to remove actual grants for each node and replace by newer based only on roles. Note that you can build other type of logic. Take the example as a basis.

- *The variable **UUID** is the folder Unique Identifier.*
- *The variable **MAX_DEPTH** sets how much deep should advance the script in subfolders children (note than 0 indicates the UUID folder).*
- *The variables **roleName** and **roleGrant** arrays are the pairs of role name and roles grants to be added.*
- *The variable **LOG_FILE_NAME** is the name of the log file saved under $TOMCAT_HOME folder.*

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.Writer;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.MessageFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

import com.openkm.bean.Permission;
import com.openkm.core.Config;

// PARAMETERS
String LOG_FILE_NAME = "Security";
int MAX_DEPTH = Integer.MAX_VALUE;
String folder_UUID = "162735a1-661c-4555-bf53-4e9115f38c62";
String[] roleName = {"ROLE_X", "ROLE_Y"};
int[] roleGrant = {Permission.READ, Permission.READ | Permission.WRITE};

public static class FileLogger {

    public static final String DATE_FORMAT = "yyyy-MM-dd HH:mm:ss,SSS";
    public static final String LEVEL_INFO = "INFO ";
    public static final String LEVEL_WARN = "WARN ";
    public static final String LEVEL_ERROR = "ERROR";

    public static void info(String baseName, String message) throws IOException {
        logWrite(baseName, LEVEL_INFO, message);
    }
```

```
    public static void logWrite(String baseName, String level, String message) throws
        Writer sLogger = new FileWriter(getLogFile(baseName), true);
        sLogger.write(getLogEntry(level, message));
        sLogger.flush();
        sLogger.close();
    }

    public static String getLogFile(String baseName) {
        String fileDate = new SimpleDateFormat("yyyyMMdd").format(new Date());
        return Config.HOME_DIR + File.separator + baseName + "_" + fileDate + ".log";
    }

    public static void info(String baseName, String message, String param1, String pa
        logWrite(baseName, LEVEL_INFO, message, param1, param2);
    }

    public static void warm(String baseName, String message, String param1, String pa
        logWrite(baseName, LEVEL_WARN, message, param1, param2);
    }

    public static void error(String baseName, String message, String param1, String p
        logWrite(baseName, LEVEL_ERROR, message, param1, param2);
    }

    public static void logWrite(String baseName, String level, String message, String
            throws IOException {
        Writer sLogger = new FileWriter(getLogFile(baseName), true);
        sLogger.write(getLogEntry(level, message, param1, param2));
        sLogger.flush();
        sLogger.close();
    }

    public static String getLogEntry(String level, String message) {
        StringBuilder sb = new StringBuilder();
        sb.append(new SimpleDateFormat(DATE_FORMAT).format(new Date()));
        sb.append(" ");
        sb.append(level);
        sb.append(" ");
        sb.append(message);
        sb.append("\n");
        return sb.toString();
    }

    public static String getLogEntry(String level, String message, String param1, Str
        Object[] params = {param1, param2};
        StringBuilder sb = new StringBuilder();
        sb.append(new SimpleDateFormat(DATE_FORMAT).format(new Date()));
        sb.append(" ");
        sb.append(level);
        sb.append(" ");
        sb.append(MessageFormat.format(message, params));
        sb.append("\n");
        return sb.toString();
    }
}

public void updateSecurity(Connection con, String UUID, int level) throws SQLExceptio
    FileLogger.info(LOG_FILE_NAME, "updateSecurity level:''{0}'' with UUID:''{1}''",
    PreparedStatement ps = null;
    ResultSet rs = null;

    if (level == 0) {
        setParentNodeSecurity(con, UUID);
        removeAllChildrenNodeSecurity(con, UUID);
        setAllChildrenNodeSecurity(con, UUID);
    }
```

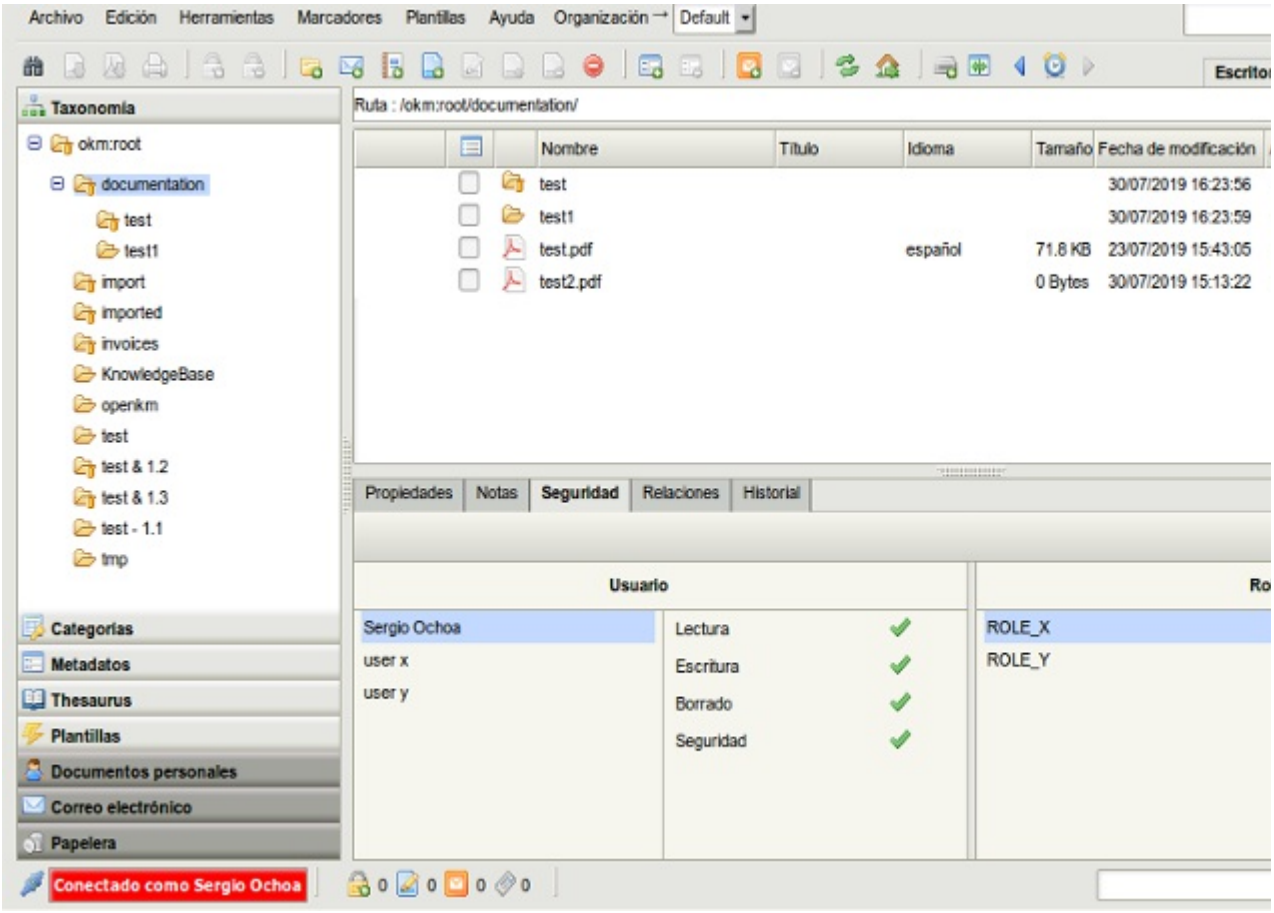
```
    if (level < MAX_DEPTH) {
        String sqlfindFolderChildren = "SELECT NBS_UUID FROM OKM_NODE_BASE WHERE NBS_

        try {
            ps = con.prepareStatement(sqlfindFolderChildren);
            ps.setString(1, UUID);
            rs = ps.executeQuery();

            while (rs.next()) {
                String childUUID = rs.getString("NBS_UUID");
                removeAllChildrenNodeSecurity(con, childUUID);
                setAllChildrenNodeSecurity(con, childUUID);
                updateSecurity(con, childUUID, level + 1);
            }
        } catch (SQLException e) {
            FileLogger.error(LOG_FILE_NAME, "updateSecurity with UUID:''{0}'' con err
        } finally {
            ps.close();
            rs.close();
        }
    }
}

public void removeAllChildrenNodeSecurity(Connection con, String UUID) throws SQLExce
    PreparedStatement ps = null;
    String deleteChildrenNodeRoles = "DELETE FROM OKM_NODE_ROLE_PERMISSION where NRP_
    String deleteChildrenNodeUsers = "DELETE FROM OKM_NODE_USER_PERMISSION where NUP_

    try {
        ps = con.prepareStatement(deleteChildrenNodeRoles);
        ps.setString(1, UUID);
        ps.executeUpdate();
        ps = con.prepareStatement(deleteChildrenNodeUsers);
        ps.setString(1, UUID);
        ps.executeUpdate();
        con.commit();
    } catch (SQLException e) {
        FileLogger.error(LOG_FILE_NAME, "removeAllChildrenNodeSecurity with UUID:''{0
                e.getMessage());
    } finally {
        ps.close();
    }
}

public void setAllChildrenNodeSecurity(Connection con, String UUID) throws SQLExcepti
    PreparedStatement ps = null;
    ResultSet rs = null;
    String sqlfindAllChildren = "SELECT NBS_UUID FROM OKM_NODE_BASE WHERE NBS_PARENT=
    String insertRoleSecurityChildren = "INSERT INTO OKM_NODE_ROLE_PERMISSION (NRP_NO

    try {
        ps = con.prepareStatement(sqlfindAllChildren);
        ps.setString(1, UUID);
        rs = ps.executeQuery();
        ps = con.prepareStatement(insertRoleSecurityChildren);

        while (rs.next()) {
            String childUUID = rs.getString("NBS_UUID");

            for (int i = 0; i < roleName.length; i++) {
                ps.setString(1, childUUID);
                ps.setInt(2, roleGrant[i]);
                ps.setString(3, roleName[i]);
                ps.addBatch();
            }
        }
```

```
        ps.executeBatch();
        con.commit();
    } catch (SQLException e) {
        FileLogger.error(LOG_FILE_NAME, "removeAllChildrenNodeSecurity with UUID:''{0
                e.getMessage());
    } finally {
        ps.close();
        rs.close();
    }
}

public void setParentNodeSecurity(Connection con, String UUID) throws SQLException, I
    PreparedStatement ps = null;
    String deleteChildrenNodeRoles = "DELETE FROM OKM_NODE_ROLE_PERMISSION where NRP_
    String deleteChildrenNodeUsers = "DELETE FROM OKM_NODE_USER_PERMISSION where NUP_
    String insertRoleSecurityChildren = "INSERT INTO OKM_NODE_ROLE_PERMISSION (NRP_NO

    try {
        ps = con.prepareStatement(deleteChildrenNodeRoles);
        ps.setString(1, UUID);
        ps.executeUpdate();
        ps = con.prepareStatement(deleteChildrenNodeUsers);
        ps.setString(1, UUID);
        ps.executeUpdate();
        ps = con.prepareStatement(insertRoleSecurityChildren);

        for (int i = 0; i < roleName.length; i++) {
            ps.setString(1, UUID);
            ps.setInt(2, roleGrant[i]);
            ps.setString(3, roleName[i]);
            ps.addBatch();
        }

        ps.executeBatch();
        con.commit();
    } catch (SQLException e) {
        FileLogger.error(LOG_FILE_NAME, "setParentNodeSecurity with UUID:''{0}'' con
    } finally {
        ps.close();
    }
}


try {
    FileLogger.info(LOG_FILE_NAME, "**** Security change started ****");
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = null;
    con = DriverManager.getConnection("jdbc:mysql://localhost:3306/okmdb", "openkm",
    con.setAutoCommit(false);
    updateSecurity(con, folder_UUID, 0);
    con.close();
    FileLogger.info(LOG_FILE_NAME, "**** Security change ended ****");
} catch (Exception e) {
    print(e.getMessage());
}
```

After the script execution we must reindex Lucene indexes because the security changes has been done at the database level and it is necessary for maintaining integrity with search engine.

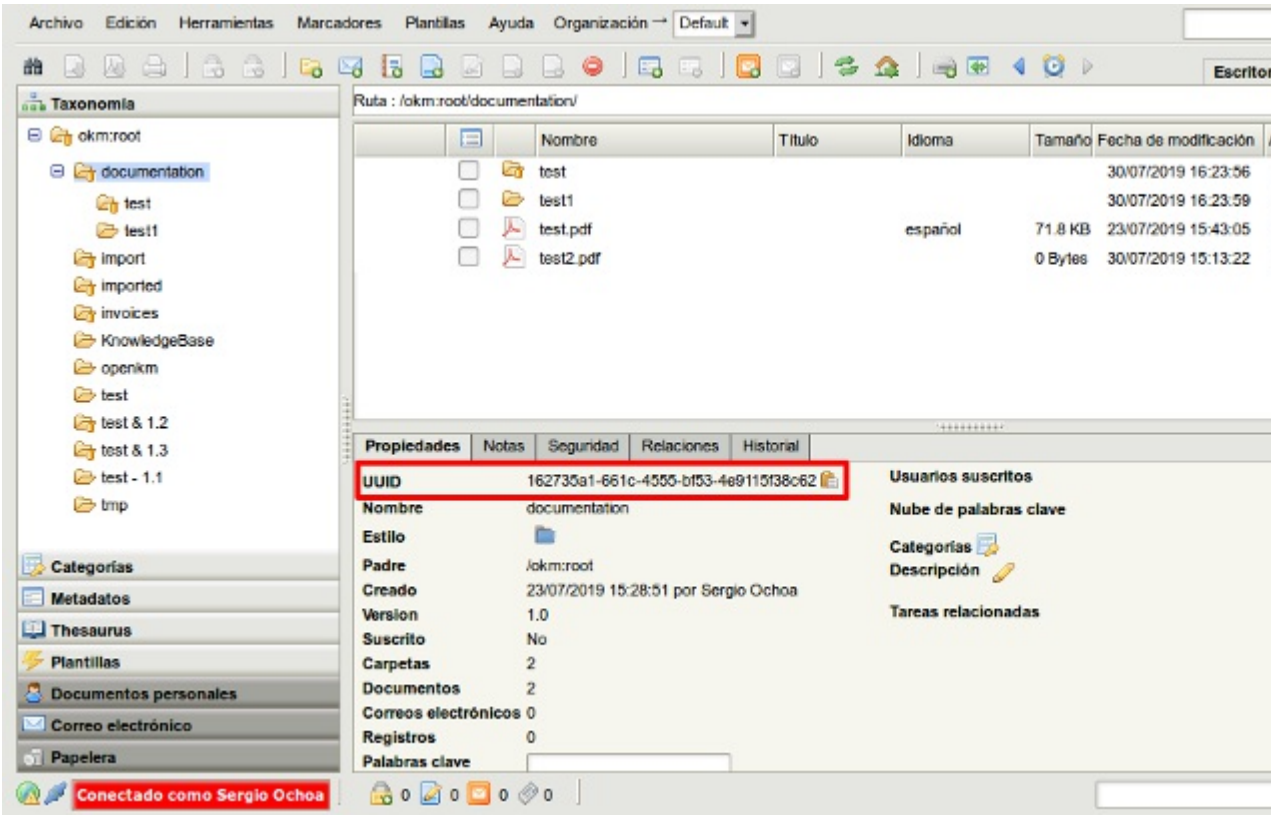
Images

Initial security status



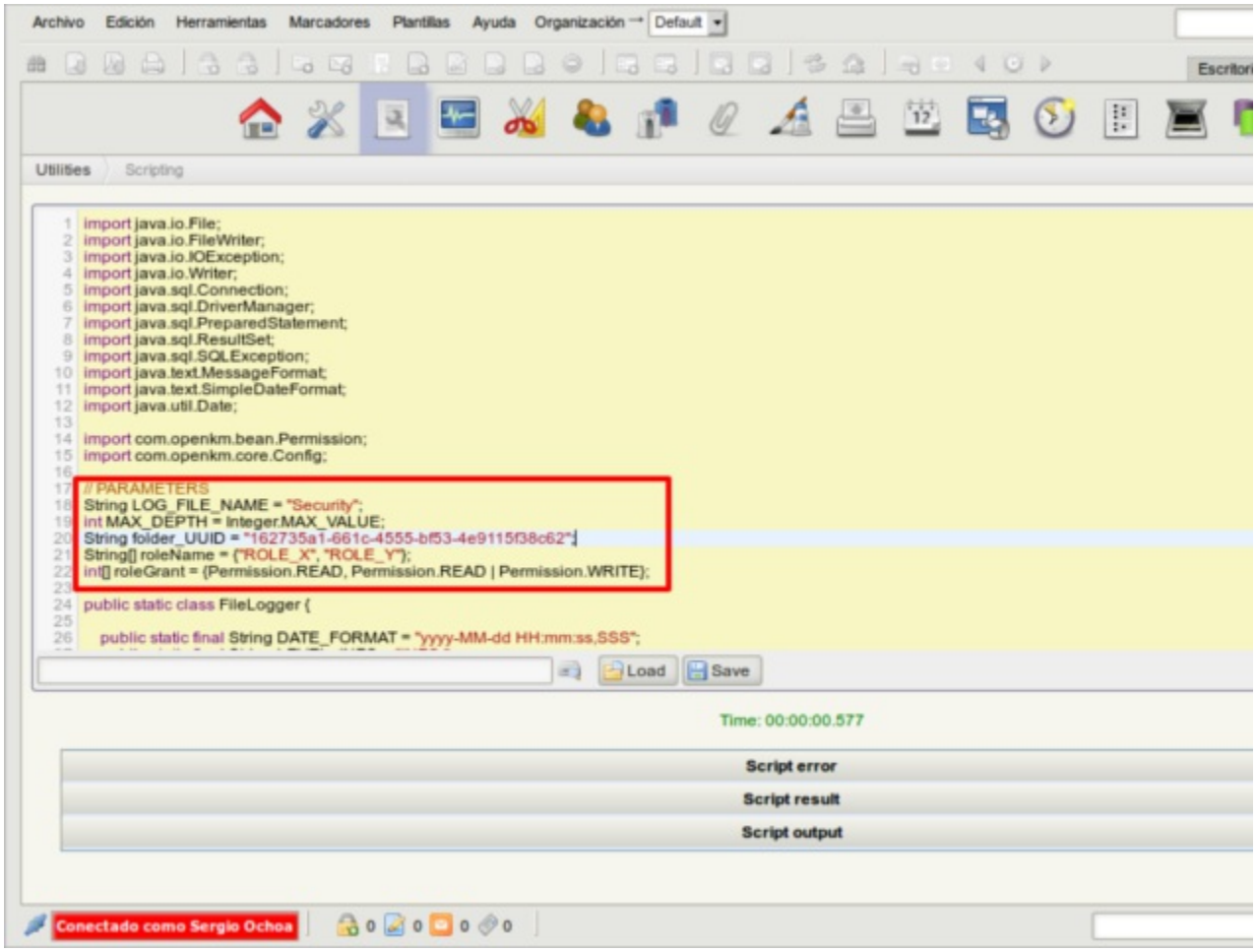
UUID of the folder

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Execution

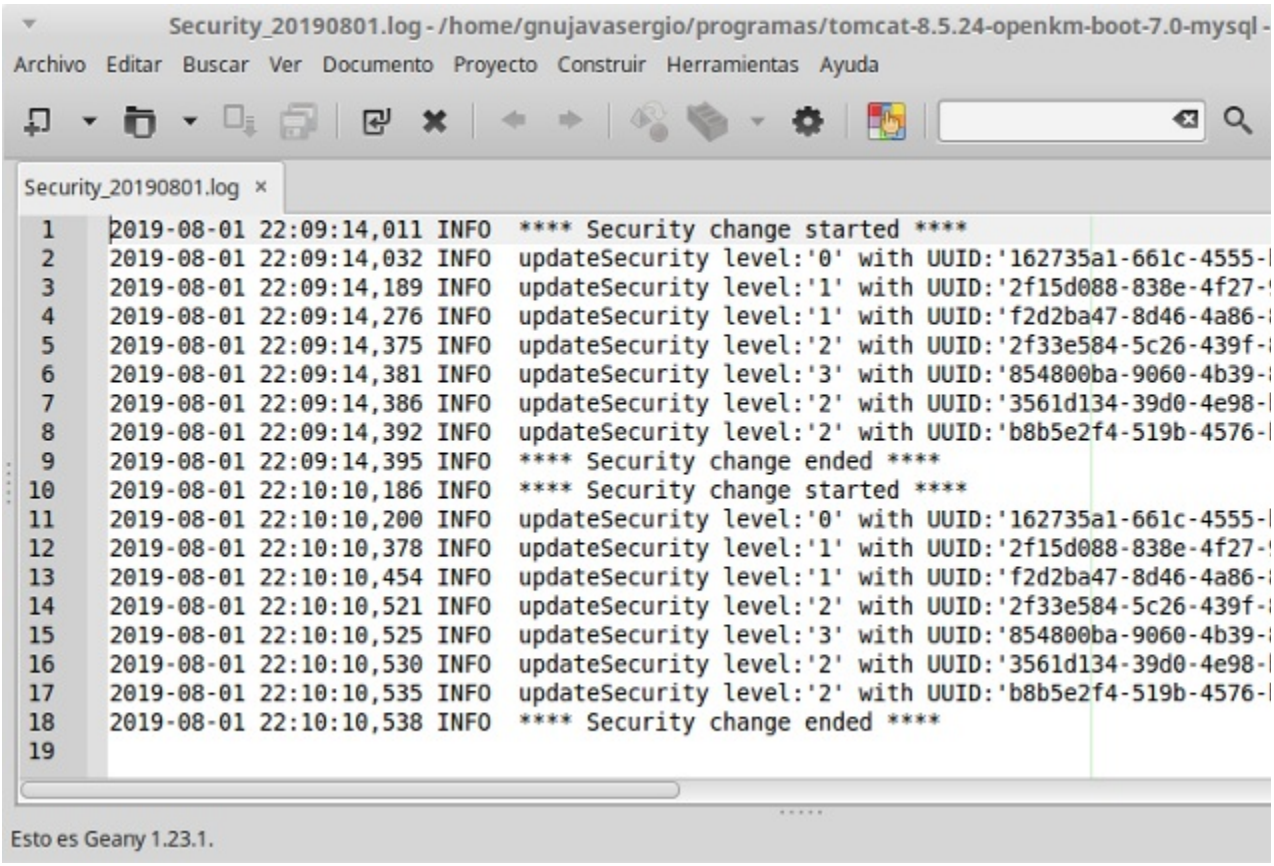
Change script parameters by your owns.



Log

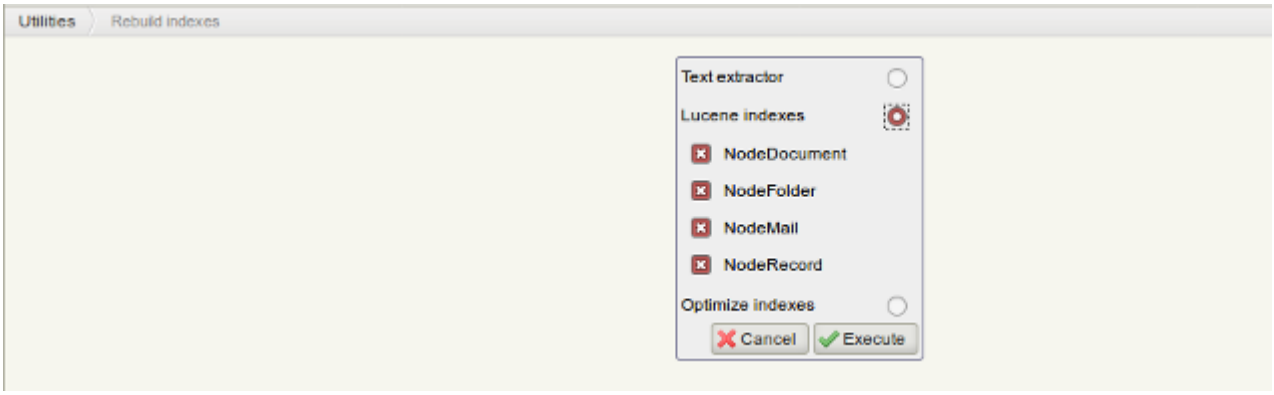
Consider take a look at file log to see progress of the process and check for errors.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

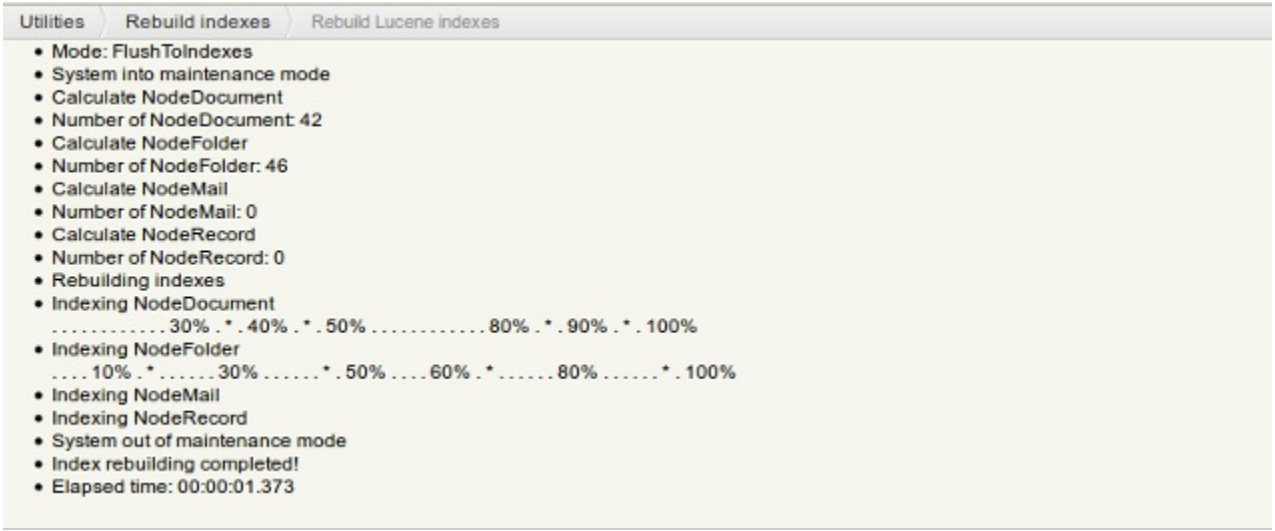


Rebuild Lucene indexes

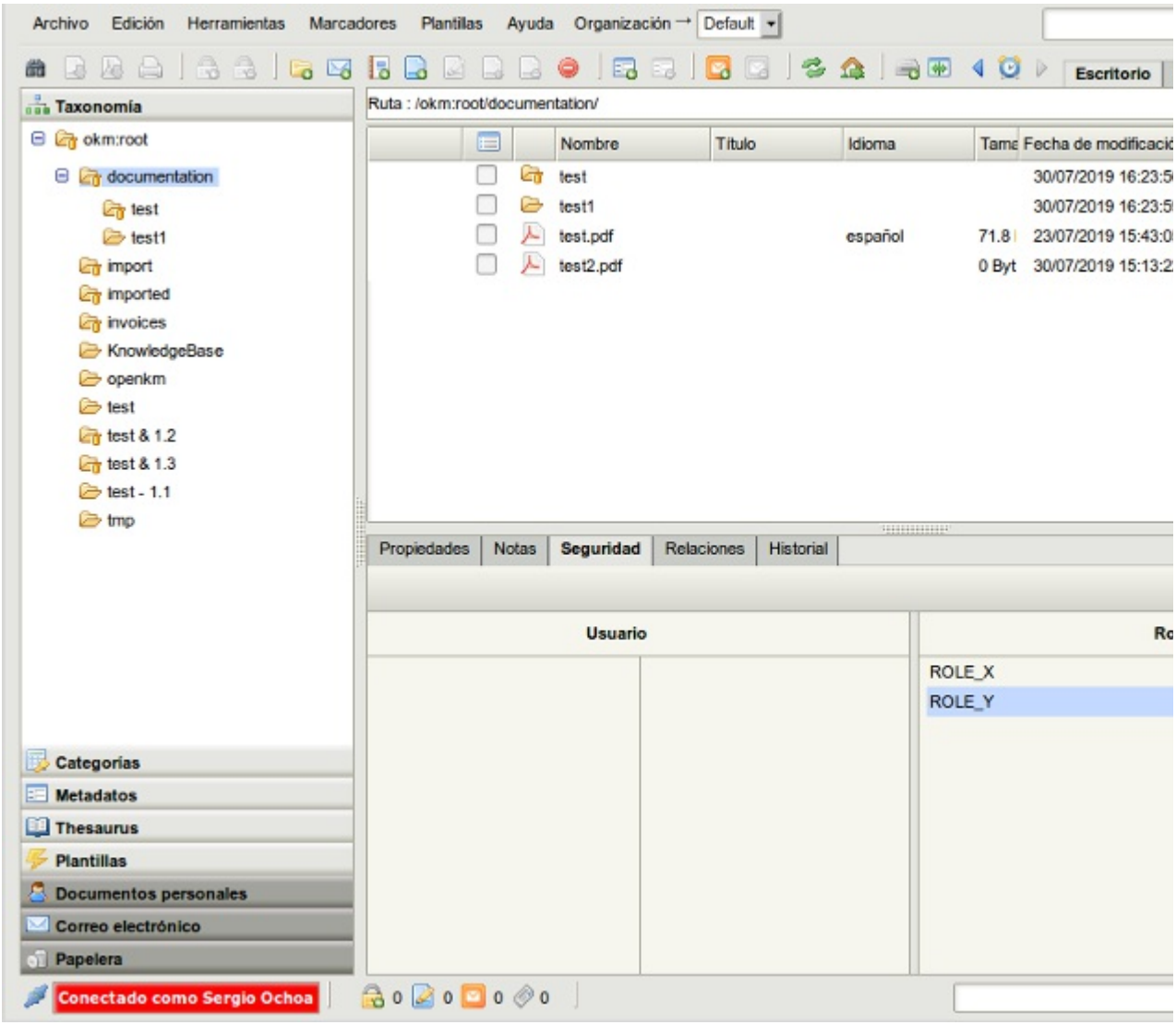
Caused by changes of the security at a database level, it is necessary for maintaining integrity with search engine, reindex Lucene indexes.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Security changed applied



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script samples for automation

Script	*Description*
Script for automation - Execute OMR	*Executes the OMR engine for each new uploaded image.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Script for automation - Execute OMR

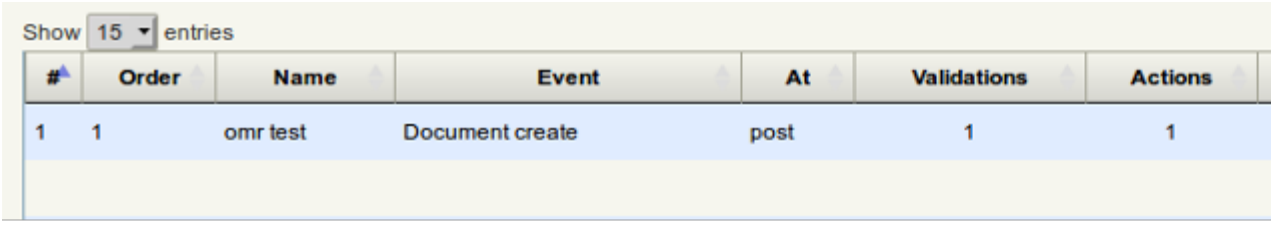
The script process an image with OMR engine from Automation each time new document is uploaded. With minimal changes can be used in conjunction with crontab.

> *The value of the variable **omId** is the **template id**.*

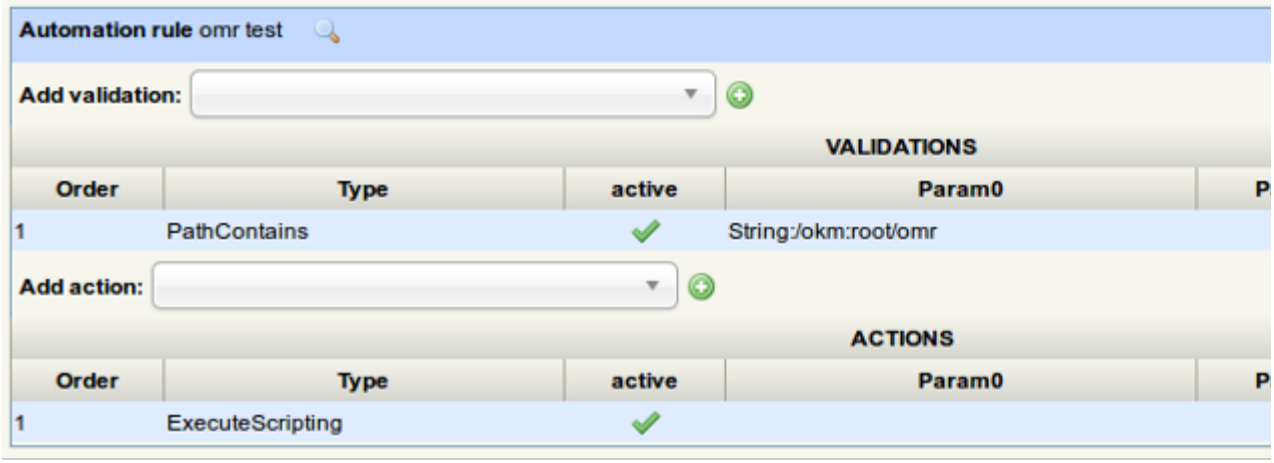
```
import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.omr.OMRHelper;
import com.openkm.util.ContextWrapper;

try {
    String token = DbSessionManager.getInstance().getSystemToken();

    OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument.class);
    OMRHelper omrHelper = ContextWrapper.getContext().getBean(OMRHelper.class);

    long omId = 1;
    Document doc = okmDocument.getProperties(token, uuid);

    if (omrHelper.isValid(doc)) {
        omrHelper.processAndStoreMetadata(omId, uuid);
    }
} catch (Exception e) {
    System.out.println(e.getMessage());
}
```

Images

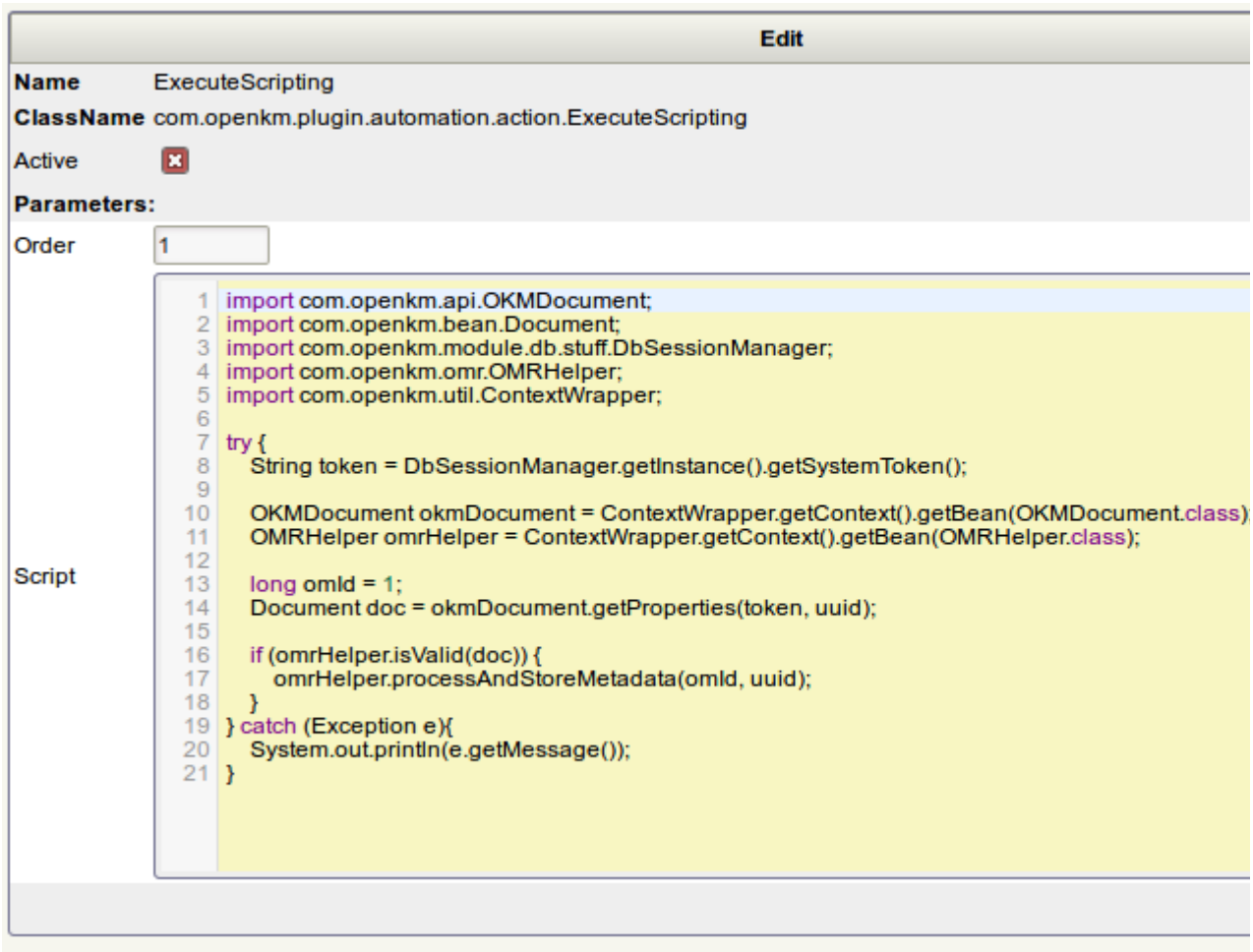
Register automation rule



With some PathContains validation and ExecuteScripting action.



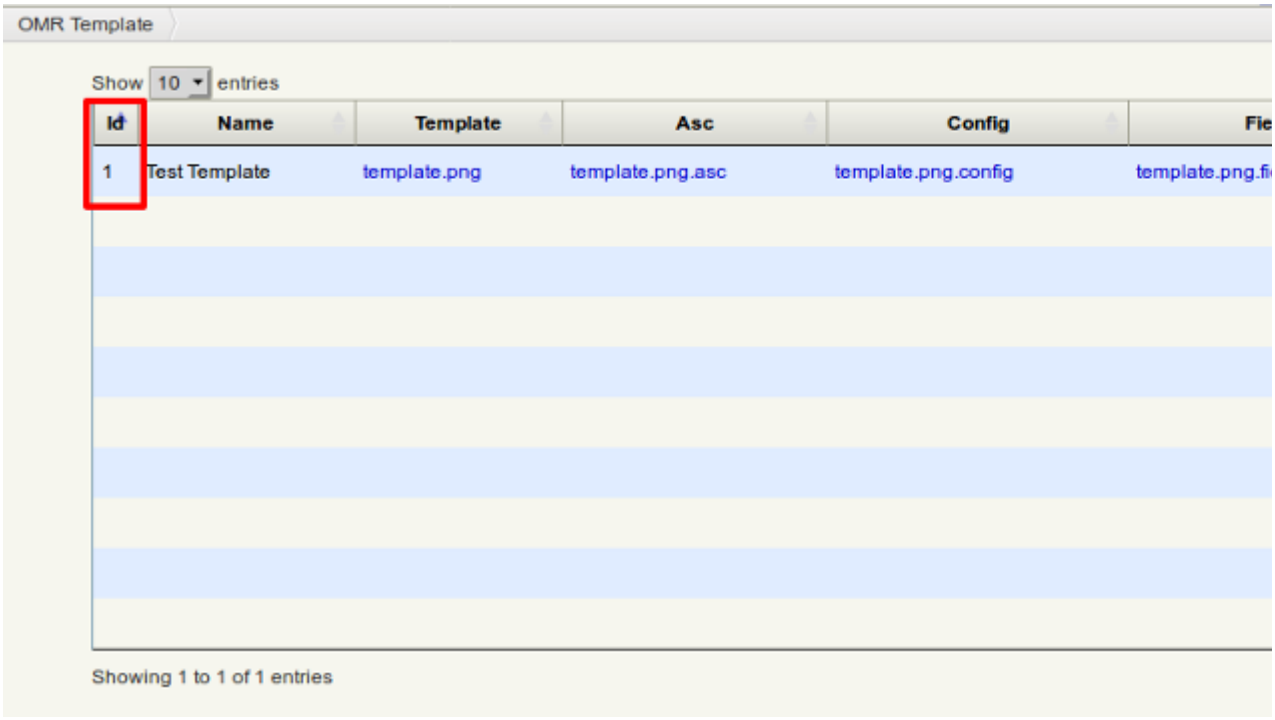
The script code:



Templates defintion

*Get the **omId value** of the **template**.*

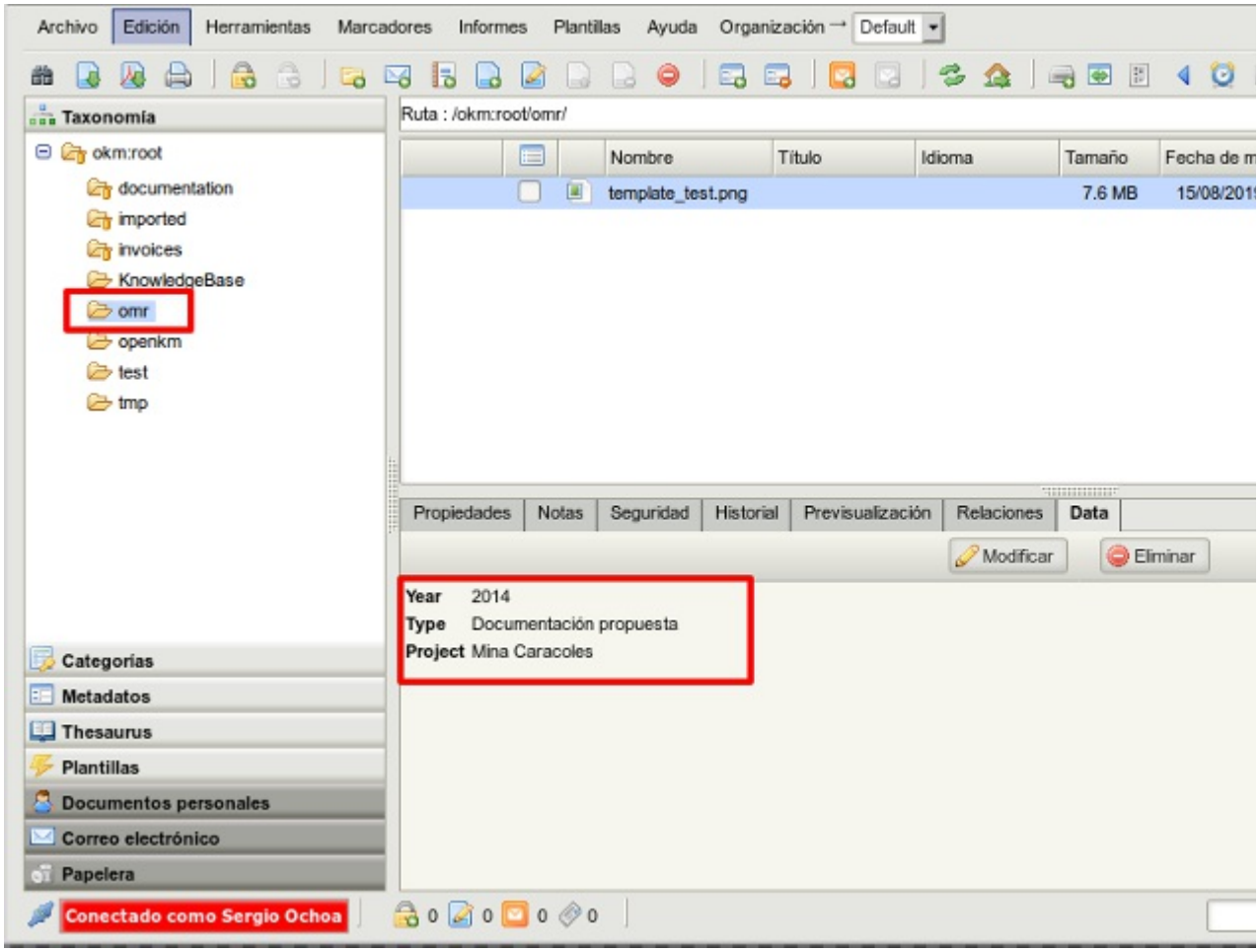
Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Result

Each document uploaded into folder path (or subfolders), when is valid image file, is processed by OMR engine.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Crontab samples

Script	*Description*
Crontab sample - Basic file system document importer	*The script imports files from the server file system.*
Crontab sample - Document importer with metadata values	*Imports files with metadata values from the server file system.*
Crontab task - XML importer	*Import files with metadata values from XML files.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Crontab sample - Basic file system document importer

The script import files from file system of the server.

- *Files are imported from the server file system.*
- *Files are periodically imported by a crontab task to a OpenKM folder (/okm:root/Scans).*

```
package com.openkm.plugin.cron;

import java.io.File;
import java.io.FileInputStream;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.core.ItemExistsException;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.util.ContextWrapper;

import net.xeoh.plugins.base.annotations.PluginImplementation;

@PluginImplementation
public class SystemDocumentImporter extends BaseCronPlugin implements CronAdapter {

    private Logger log = LoggerFactory.getLogger(SystemDocumentImporter.class);

    private OKMDocument document = ContextWrapper.getContext().getBean(OKMDocument.cl
    private OKMFolder folder = ContextWrapper.getContext().getBean(OKMFolder.class);
    private String token = DbSessionManager.getInstance().getSystemToken();

    @Override
    public String getName() {
        return "System Document Importer";
    }

    @Override
    public String getCronExpression() {
        return "0 5 * * * *";
    }

    @Override
    public void execute() {
        String okmPath = "/okm:root/Scans/";
        File file = new File("/home/opemkm/import");
        autoImport(okmPath, file);
    }

    private void autoImport(String okmPath, File fldpath) {
        try {
            log.info("Scanning " + fldpath.getName());
            for (File file : fldpath.listFiles()) {
                log.info("Importing " + file.getName());
                try {
                    if (file.isDirectory()) {
                        try {
                            folder.createSimple(token, okmPath + file.getName());
                        } catch (ItemExistsException ie) {
```

```
                            log.error("folder already exists<br>");
                            // Folder already exists - just ignore exception
                        }
                        autoImport(okmPath + file.getName() + "/", file);
                    } else {
                        // Check if file is still being written to
                        long length = file.length();
                        Thread.sleep(1000);
                        if (file.length() > length) {
                            continue; // Skip file this time
                        }
                        FileInputStream fis = new FileInputStream(file);
                        document.createSimple(token, okmPath + file.getName(), fis);
                        fis.close();
                    }
                    log.info("Created " + okmPath + file.getName());
                } catch (Exception e) {
                    log.info("Exception:" + e);
                    // Something bad happened to prevent import. Skip to next file.
                    continue;
                }
                file.delete();
            }
        } catch (Exception e) {
            log.error(e.getMessage());
        }
    }

}
```

Register a new plugin

- *To install new plugin **create a jar file** and **copy into your $TOMCAT/plugins** folder. More information at Creating a JAR file with Eclipse.*
- *Go to **Administration > Utilities > Plugins** and click at top right the Reload plugins button.*
- *In **plugins table will be shown** the new plugin.*

Images

Crontab task

Crontab

Show 15 entries

Name	Expression	Class Name	Last Begin	
Antivirus Checker Worker	0 */5 * * * *	com.openkm.plugin.cron.AntivirusChecker	18-08-2019 11:45:00	18-
Check Disk Space	0 0 0 * * *	com.openkm.plugin.cron.CheckDiskSpace	18-08-2019 00:00:00	18-
Check Repository Integrity	0 0 0 * * SAT	com.openkm.plugin.cron.CheckRepositoryIntegrity		
Dashboard Activity Purge	0 0 0 * * *	com.openkm.plugin.cron.DashboardActivity	18-08-2019 00:00:00	18-
Document Importer With Metadata	0 5 * * * *	com.openkm.plugin.cron.DocumentImporterWithMetadata	18-08-2019 11:27:28	18-
Process Pending Task	0 */5 * * * *	com.openkm.plugin.cron.ProcessPendingTask	18-08-2019 11:45:00	18-
Purge Empty Datastore Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyDatastoreDirectories	18-08-2019 00:00:00	18-
Purge Empty Extraction Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyExtractionDirectories	18-08-2019 00:00:00	18-
Purge Old Trash Nodes	0 0 0 * * *	com.openkm.plugin.cron.PurgeOldTrashesNodes	18-08-2019 00:00:00	18-
Purge Temporal Printing Files	0 0 0 * * *	com.openkm.plugin.cron.PurgeTemporalPrintingFiles	18-08-2019 00:00:01	18-
Purge Tomcat Logs	0 0 0 * * *	com.openkm.plugin.cron.PurgeTomcatLogs	18-08-2019 00:00:00	18-
Repository Info	0 0 0 * * *	com.openkm.plugin.cron.RepositoryInfo	18-08-2019 00:00:00	18-
Send Grouped Notifications	0 0 0 * * *	com.openkm.plugin.cron.SendGroupedNotifications	18-08-2019 00:00:00	18-
System Document Importer	0 5 * * * *	com.openkm.plugin.cron.SystemDocumentImporter	18-08-2019 02:05:02	18-
Text Extractor Issues	0 0 0 * * *	com.openkm.plugin.cron.TextExtractorIssues	18-08-2019 00:00:00	18-

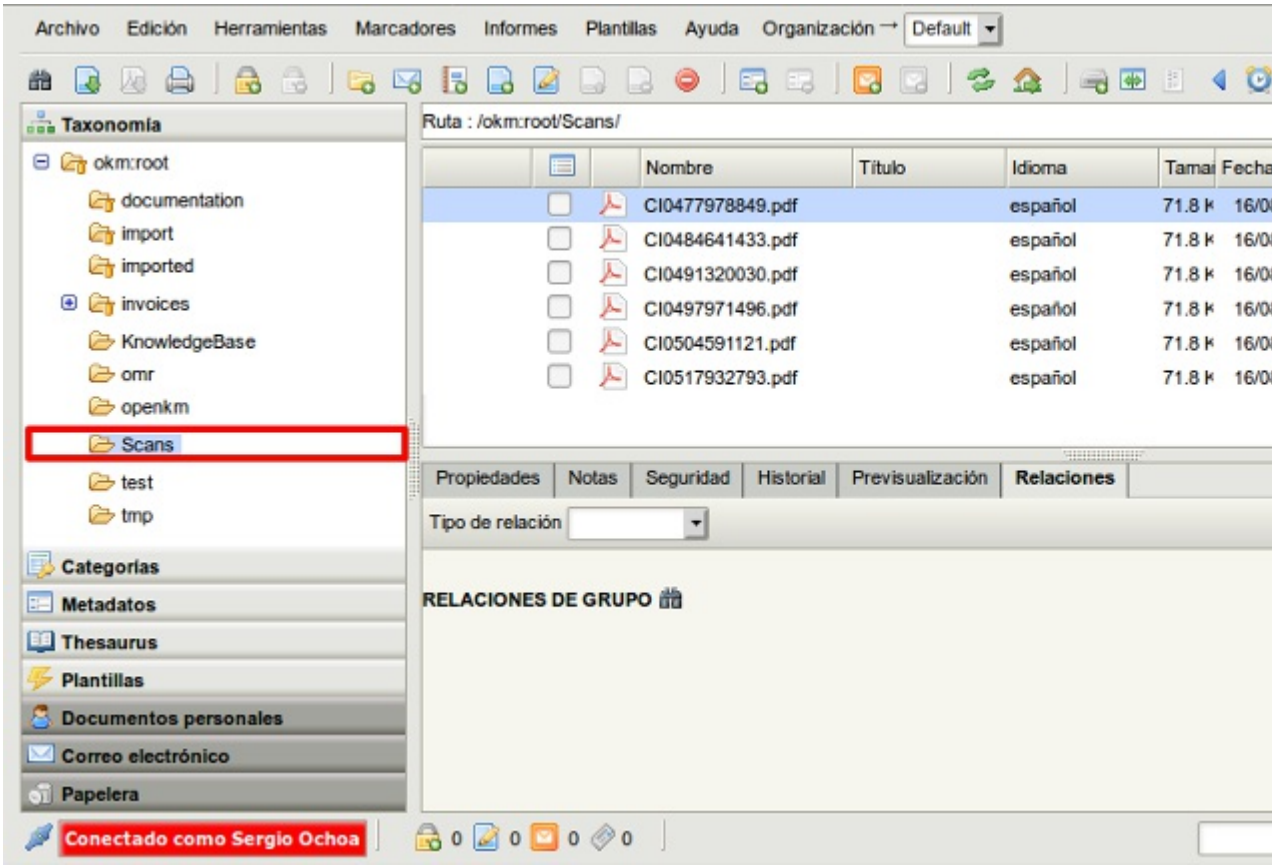
Showing 1 to 15 of 17 entries

Files in the application server

CI0477978849.pdf	244,2 KB Adobe Acrobat Document	29/09/11
CI0484641433.pdf	270,3 KB Adobe Acrobat Document	29/09/11
CI0491320030.pdf	276,4 KB Adobe Acrobat Document	29/09/11
CI0497971496.pdf	279,0 KB Adobe Acrobat Document	29/09/11
CI0504591121.pdf	278,9 KB Adobe Acrobat Document	29/09/11
CI0517932793.pdf	305,1 KB Adobe Acrobat Document	29/09/11

Result

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


Archivo
Edición
Herramientas
Marcadores
Informes
Plantillas
Ayuda
Organización
Default
Taxonomía
okm:root
documentation
import
imported
invoices
KnowledgeBase
omr
openkm
Scans
test
tmp
Categorías
Metadatos
Thesaurus
Plantillas
Documentos personales
Correo electrónico
Papelera
Conectado como Sergio Ochoa
Ruta : /okm:root/Scans/
Nombre
Título
Idioma
Tamaı
Fecha
CI0477978849.pdf
CI0484641433.pdf
CI0491320030.pdf
CI0497971496.pdf
CI0504591121.pdf
CI0517932793.pdf
español
71.8 k
Propiedades
Notas
Seguridad
Historial
Previsualización
Relaciones
Tipo de relación
RELACIONES DE GRUPO

Crontab sample - Document importer with metadata values

The script imports files with metadata values from the file system of the server.

- *Files stored in the OpenKM server have a file name format like some_text - (CUPS,REN).pdf*
- *The variable **grpName** is the metadata group to be inserted.*
- *The variable **contractUUID** is the UUID of the OpenKM where the files will be imported.*
- *The variable **systemFolder** is the file system path.*
- *Verifies paths and extracts metadata from text between the characters "**(**" "**)**" present in the file name.*
- *Files are stored at basePath/year/month (when year or month folders are not present are automatically created).*
- *At the end, it send a mail with import results.*

Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.1//EN"
"http://www.openkm.com/dtd/property-groups-2.1.dtd">
<property-groups>
    <property-group label="Contract" name="okg:contract">
    <input label="Cups" type="text" name="okp:contract.cups" width="200px"/>
    <input label="Año" type="text" name="okp:contract.year" width="200px"/>
    <select label="Mes" name="okp:contract.month" type="simple">
        <option label="Enero" value="enero"/>
     <option label="Febrero" value="febrero"/>
     <option label="Marzo" value="marzo"/>
        <option label="Abril" value="abril"/>
        <option label="Mayo" value="mayo"/>
        <option label="Junio" value="junio"/>
        <option label="Julio" value="julio"/>
        <option label="Agosto" value="agosto"/>
        <option label="Septiembre" value="septiembre"/>
        <option label="Octubre" value="octubre"/>
        <option label="Noviembre" value="noviembre"/>
        <option label="Diciembre" value="diciembre"/>
    </select>
  </property-group>
</property-groups>
```

The script:

```
package com.openkm.plugin.cron;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.util.ArrayList;
import java.util.Calendar;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.apache.cxf.bus.extension.ExtensionException;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import com.google.gson.Gson;
import com.openkm.api.OKMDocument;
import com.openkm.api.OKMFolder;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.api.OKMRepository;
import com.openkm.bean.Document;
import com.openkm.core.AccessDeniedException;
import com.openkm.core.Config;
import com.openkm.core.DatabaseException;
import com.openkm.core.ItemExistsException;
import com.openkm.core.PathNotFoundException;
import com.openkm.core.PromotedAsRecordLockException;
import com.openkm.core.RepositoryException;
import com.openkm.core.UnsupportedMimeTypeException;
import com.openkm.core.VirusDetectedException;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.plugin.automation.AutomationException;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.MailUtils;

import net.xeoh.plugins.base.annotations.PluginImplementation;

@PluginImplementation
public class DocumentImporterWithMetadata extends BaseCronPlugin implements CronAdapt

    private Logger log = LoggerFactory.getLogger(DocumentImporterWithMetadata.class);

    @Autowired
    private OKMDocument okmDocument;

    @Autowired
    private OKMFolder okmFolder;

    @Autowired
    private OKMRepository okmRepository;

    @Autowired
    private OKMPropertyGroup okmPropertyGroup;

    @Autowired
    private MailUtils mailUtils;

    @Autowired
    private FileUtils fileUtils;

    private String token = DbSessionManager.getInstance().getSystemToken();
    private List<Status> bad = new ArrayList<>();
    private List<Status> good = new ArrayList<>();
    private List<String> toAddress = new ArrayList<>();

    @Override
    public String getName() {
        return "Document Importer With Metadata";
    }

    @Override
    public String getCronExpression() {
```

```
        return "0 5 * * * *";
    }

    @Override
    public void execute() {
        try {
            String msg = documentImporterWithMetadata();

            if (msg != null && !msg.isEmpty()) {
                // Sending mails
                toAddress.add("gdgsucre@mail.com");
                mailUtils.sendMessage(toAddress, "Importing report", msg);
                // Or send mail admin user
                mailUtils.sendCronTaskResult(getName(), msg);
            }
        } catch (Exception e) {
            log.error(e.getMessage());
        }
    }

    private String documentImporterWithMetadata()
            throws AccessDeniedException, PathNotFoundException, RepositoryException,
            UnsupportedEncodingException, ItemExistsException, AutomationException, P
        // Months
        List<String> month = new ArrayList<>();
        month.add("ENERO");
        month.add("FEBRERO");
        month.add("MARZO");
        month.add("ABRIL");
        month.add("MAYO");
        month.add("JUNIO");
        month.add("JULIO");
        month.add("AGOSTO");
        month.add("SEPTIEMBRE");
        month.add("OCTUBRE");
        month.add("NOVIEMBRE");
        month.add("DICIEMBRE");

        String grpName = "okg:contract";
        String contractUUID = "99bd0a50-fc98-409d-9b92-62528c42707f";
        String systemFolder = "/home/openkm/pending_to_import_folder";
        String basePath = okmRepository.getNodePath(token, contractUUID);
        Gson gson = new Gson();

        // Loading files
        File files = new File(systemFolder);
        File[] listOfFiles = files.listFiles();

        for (int i = 0; i < listOfFiles.length; i++) {
            File file = listOfFiles[i];
            String name = file.getName();
            String extension = fileUtils.getFileExtension(name);
            if (file.isFile() && extension.equals("pdf")) {
                Status status = new Status();
                status.fileName = name;
                if (status.fileName.indexOf("(") > 0 && status.fileName.indexOf("(")
                    // Get metadata
                    String metadata = status.fileName.substring((status.fileName.inde
                            status.fileName.lastIndexOf(")"));

                    // get data
                    String[] data = metadata.split(",");
                    if (data.length != 2) {
                        status.error = "Incorrect format -> name (cups,REN)<br>";
                    } else {
                        status.cups = data[0].toUpperCase().replaceAll(" ", "");
                        Calendar calendar = Calendar.getInstance();
```

```
                    calendar.setTimeInMillis(file.lastModified());
                    status.year = String.valueOf(calendar.get(Calendar.YEAR));
                    status.month = month.get(calendar.get(Calendar.MONTH));

                    if (!data[1].toUpperCase().equals("REN")) {
                        status.error = "Error type REN not found<br>";
                    }

                    if (status.error.equals("")) {
                        // test if folder year exists otherside create it
                        String path = basePath + "/" + status.year;

                        if (!okmRepository.hasNode(token, path)) {
                            okmFolder.createSimple(token, path);
                        }

                        // test if folder month exists otherside create it
                        path = path + "/" + status.month;

                        if (!okmRepository.hasNode(token, path)) {
                            okmFolder.createSimple(token, path);
                        }

                        // Create document and adding metadata
                        try {
                            // Removing extra ( contents )
                            String fileName = status.fileName.substring(0, status

                            // Removing - and spaces at ends
                            while (fileName.substring(fileName.length() - 1).equa
                                    || fileName.substring(fileName.length() - 1).
                                fileName = fileName.substring(0, fileName.length(
                            }

                            fileName = fileName + ".pdf";
                            path = path + "/" + fileName;
                            long length = file.length();
                            Thread.sleep(1000);
                            if (file.length() > length) {
                                continue; // Skip file this time
                            }
                            FileInputStream fis = new FileInputStream(file);
                            Document doc = okmDocument.createSimple(token, path,
                            status.dstPath = path;
                            fis.close();

                            // Create new metadata
                            Map<String, String> properties = new HashMap<>();
                            properties.put("okp:contract.cups", status.cups);
                            properties.put("okp:contract.year", status.year);
                            properties.put("okp:contract.month", gson.toJson(new

                            okmPropertyGroup.addGroup(token, doc.getPath(), grpNa

                            // Delete file
                            boolean success = file.delete();
                            if (!success) {
                                status.error += "File can not been deleted";
                            }
                        } catch (PathNotFoundException e) {
                            status.error += "PathNotFoundException";
                        } catch (ItemExistsException e) {
                            status.error += "ItemExistsException";
                        } catch (UnsupportedMimeTypeException e) {
                            status.error += "UnsupportedMimeTypeException";
                        } catch (VirusDetectedException e) {
                            status.error += "VirusDetectedException";
```

```
                            } catch (RepositoryException e) {
                                status.error += "RepositoryException";
                            } catch (DatabaseException e) {
                                status.error += "DatabaseException";
                            } catch (ExtensionException e) {
                                status.error += "ExtensionException";
                            } catch (IOException e) {
                                status.error += "IOException";
                            } catch (Exception e) {
                                status.error += e.getMessage();
                            }
                        }
                    }
                } else {
                    status.error = "Document format incorrect -> nombre - (cups,REN)<
                }

                if (!status.error.equals("")) {
                    bad.add(status);
                } else {
                    good.add(status);
                }
            }
        }

        StringBuffer result = new StringBuffer();
        result.append("<h1>Import report</h1>");
        result.append("</br></br>");
        result.append("<table boder=\"0\" cellpadding=\"2\" cellspacing=\"0\" width=\
        result.append("<tr>");
        result.append("<td bgcolor=\"silver\"><b>Name</b></td>");
        result.append("<td bgcolor=\"silver\"><b>CUPS</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Año</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Mes</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Error</b></td>");
        result.append("</tr>");
        result.append("<tr>");
        result.append("<td colspan=\"6\" bgcolor=\"silver\"><b>Errors:" + bad.size()
        result.append("</tr>");

        for (Status status : bad) {
            result.append("<tr>");
            result.append("<td>" + status.fileName + "</td>");
            result.append("<td>" + status.cups + "</td>");
            result.append("<td>" + status.year + "</td>");
            result.append("<td>" + status.month + "</td>");
            result.append("<td><font color=\"red\">" + status.error + "</font></td>")
            result.append("</tr>");
        }

        result.append("</table>");
        result.append("</br></br>");
        result.append("<table boder=\"0\" cellpadding=\"2\" cellspacing=\"0\" width=\
        result.append("<tr>");
        result.append("<td bgcolor=\"silver\"><b>Name</b></td>");
        result.append("<td bgcolor=\"silver\"><b>CUPS</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Year</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Month</b></td>");
        result.append("<td bgcolor=\"silver\"><b>Destination</b></td>");
        result.append("</tr>");
        result.append("<tr>");
        result.append("<td colspan=\"6\" bgcolor=\"silver\"><b>Imported correctly:" +
        result.append("</tr>");

        for (Status status : good) {
            result.append("<tr>");
            result.append("<td>" + status.fileName + "</td>");
```

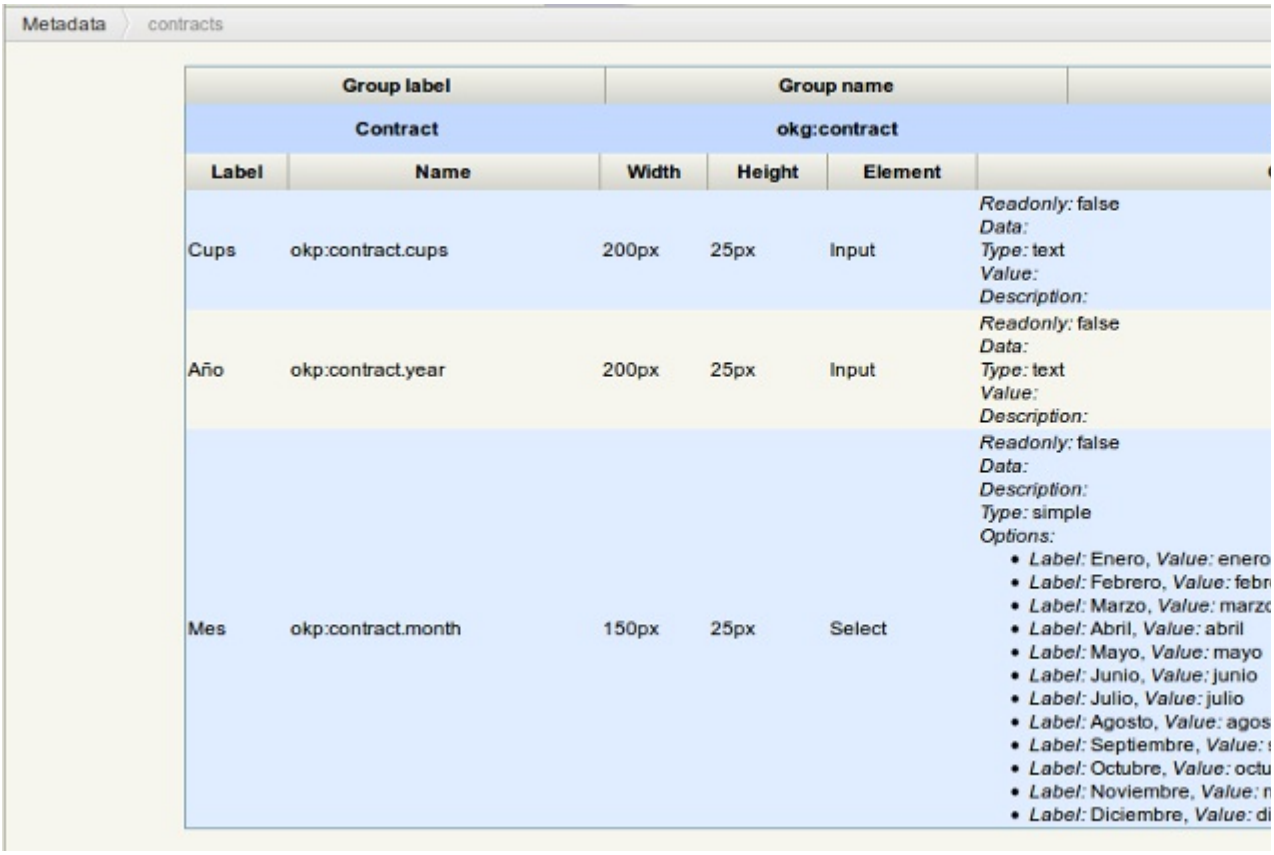
```
            result.append("<td>" + status.cups + "</td>");
            result.append("<td>" + status.year + "</td>");
            result.append("<td>" + status.month + "</td>");
            result.append("<td><a href=\"" + Config.APPLICATION_URL + "?docPath=" + U
                    + "\">" + status.dstPath + "</font></td>");
            result.append("</tr>");
        }

        result.append("</table>");

        return result.toString();
    }

    private class Status {

        public String fileName = "";
        public String dstPath = "";
        public String cups = "";
        public String month = "";
        public String year = "";
        public String error = "";
    }
}
```

Register a new plugin

- *To install new plugin **create a jar file** and **copy into your $TOMCAT/plugins** folder. More information at Creating a JAR file with Eclipse.*
- *Go to **Administration > Utilities > Plugins** and click at top right the* *Reload plugins button.*
- *In **plugins table will be shown** the new plugin.*

Images

Register metadata definition


Metadata contracts

Group label		Group name			
Contract		okg:contract			
Label	**Name**	**Width**	**Height**	**Element**	
Cups	okp:contract.cups	200px	25px	Input	*Readonly:* false *Data:* *Type:* text *Value:* *Description:*
Año	okp:contract.year	200px	25px	Input	*Readonly:* false *Data:* *Type:* text *Value:* *Description:*
Mes	okp:contract.month	150px	25px	Select	*Readonly:* false *Data:* *Description:* *Type:* simple *Options:* • *Label:* Enero, *Value:* enero • *Label:* Febrero, *Value:* febr • *Label:* Marzo, *Value:* marzo • *Label:* Abril, *Value:* abril • *Label:* Mayo, *Value:* mayo • *Label:* Junio, *Value:* junio • *Label:* Julio, *Value:* julio • *Label:* Agosto, *Value:* agos • *Label:* Septiembre, *Value:* • *Label:* Octubre, *Value:* octu • *Label:* Noviembre, *Value:* • *Label:* Diciembre, *Value:* di

Crontab task

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Crontab

Show 15 entries

Name	Expression	Class Name	Last Begin	
Antivirus Checker Worker	0 */5 * * * *	com.openkm.plugin.cron.AntivirusChecker	18-08-2019 11:45:00	18-
Check Disk Space	0 0 0 * * *	com.openkm.plugin.cron.CheckDiskSpace	18-08-2019 00:00:00	18-
Check Repository Integrity	0 0 0 * * SAT	com.openkm.plugin.cron.CheckRepositoryIntegrity		
Dashboard Activity Purge	0 0 0 * * *	com.openkm.plugin.cron.DashboardActivity	18-08-2019 00:00:00	18-
Document Importer With Metadata	0 5 * * * *	com.openkm.plugin.cron.DocumentImporterWithMetadata	18-08-2019 11:27:28	18-
Process Pending Task	0 */5 * * * *	com.openkm.plugin.cron.ProcessPendingTask	18-08-2019 11:45:00	18-
Purge Empty Datastore Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyDatastoreDirectories	18-08-2019 00:00:00	18-
Purge Empty Extraction Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyExtractionDirectories	18-08-2019 00:00:00	18-
Purge Old Trash Nodes	0 0 0 * * *	com.openkm.plugin.cron.PurgeOldTrashesNodes	18-08-2019 00:00:00	18-
Purge Temporal Printing Files	0 0 0 * * *	com.openkm.plugin.cron.PurgeTemporalPrintingFiles	18-08-2019 00:00:01	18-
Purge Tomcat Logs	0 0 0 * * *	com.openkm.plugin.cron.PurgeTomcatLogs	18-08-2019 00:00:00	18-
Repository Info	0 0 0 * * *	com.openkm.plugin.cron.RepositoryInfo	18-08-2019 00:00:00	18-
Send Grouped Notifications	0 0 0 * * *	com.openkm.plugin.cron.SendGroupedNotifications	18-08-2019 00:00:00	18-
System Document Importer	0 5 * * * *	com.openkm.plugin.cron.SystemDocumentImporter	18-08-2019 02:05:02	18-
Text Extractor Issues	0 0 0 * * *	com.openkm.plugin.cron.TextExtractorIssues	18-08-2019 00:00:00	18-

Showing 1 to 15 of 17 entries

Files in application server

document 1 - (CUPS001025,REN).pdf	5,9 KB Adobe Acrobat Document
document 2 - (CUPS001026,REN).pdf	5,9 KB Adobe Acrobat Document
document 3 - (CUPS001046,REN).pdf	5,9 KB Adobe Acrobat Document
document 4 - (CUPS001030,REN).pdf	5,9 KB Adobe Acrobat Document
document 5 - (CUPS001045,REN).pdf	5,9 KB Adobe Acrobat Document
document 6 - (CUPS001026).pdf	5,9 KB Adobe Acrobat Document
document 7 - (CUPS001046,REN).pdf	5,9 KB Adobe Acrobat Document
document 8 - CUPS001030 REC.pdf	5,9 KB Adobe Acrobat Document

Result

Mail notification

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

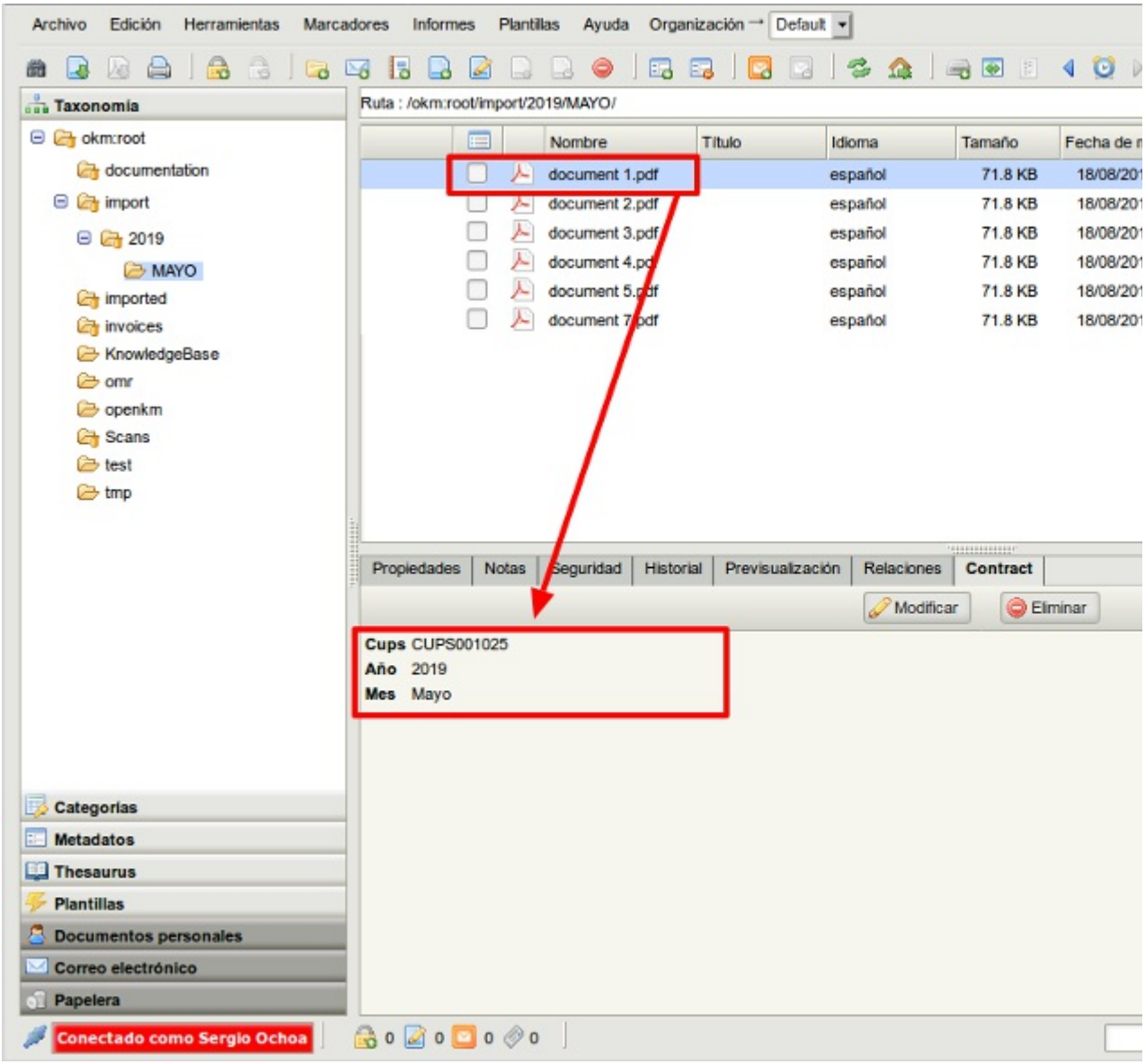
Import report

Name	CUPS	Año	Mes	Error
Errors:2				
document 8 - (CUPS001030,REC).pdf	CUPS001030	2019	MAYO	Error type RE
document 6 - (CUPS001026).pdf				Incorrect for

Name	CUPS	Year	Month	Destination
Imported correctly:6				
document 3 - (CUPS001046,REN).pdf	CUPS001046	2019	MAYO	/okm:root/import/201
document 2 - (CUPS001026,REN).pdf	CUPS001026	2019	MAYO	/okm:root/import/201
document 7 - (CUPS001046,REN).pdf	CUPS001046	2019	MAYO	/okm:root/import/201
document 4 - (CUPS001030,REN).pdf	CUPS001030	2019	MAYO	/okm:root/import/201
document 5 - (CUPS001045,REN).pdf	CUPS001045	2019	MAYO	/okm:root/import/201
document 1 - (CUPS001025,REN).pdf	CUPS001025	2019	MAYO	/okm:root/import/201

Imported files into the repository

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


Archivo
Edición
Herramientas
Marcadores
Informes
Plantillas
Ayuda
Organización
Default
Taxonomía
okm:root
documentation
import
2019
MAYO
imported
invoices
KnowledgeBase
omr
openkm
Scans
test
tmp
Ruta : /okm:root/import/2019/MAYO/
Nombre
Título
Idioma
Tamaño
document 1.pdf
document 2.pdf
document 3.pdf
document 4.pdf
document 5.pdf
document 7.pdf
español
71.8 KB
Propiedades
Notas
Seguridad
Historial
Previsualización
Relaciones
Contract
Modificar
Eliminar
Cups CUPS001025
Año 2019
Mes Mayo
Categorías
Metadatos
Thesaurus
Plantillas
Documentos personales
Correo electrónico
Papelera
Conectado como Sergio Ochoa

Crontab task - XML importer

Import files with metadata values from XML files.

This is a crontab task base in Java class packaged in a JAR.

- *The files to be imported are stored at **/home/openkm/import/Output** OpenKM server. (Variable **SYSTEM_FOLDER**+Output).*
- *The XML files with metadata values are stored at **/home/openkm/import/logfile** OpenKM server. (Variable **SYSTEM_FOLDER**+logfile).*
- *The files are imported at OpenKM folder "**/okm:root/import**". (Variable **OPENKM_FOLDER**).*

The crontab does three actions:

- *First import all files from "**/home/openkm/import/Output**" and delete them from server, after has being imported.*
- *Then load and parse xml files located at "**/home/openkm/import/logfile**" and set the metadata group values of the documents.*
- *Finally the xml files are deleted.*

Metada group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.1//EN"
                                 "http://www.openkm.com/dtd/property-groups-2.1.dtd">
<property-groups>
    <property-group label="Data" name="okg:xmldata">
        <input label="Folio" name="okp:xmldata.folio" type="text"/>
        <input label="Rut" name="okp:data.rut" type="text"/>
        <input label="Placa patente" name="okp:xmldata.placa" type="text"/>
        <input label="Nombre" name="okp:xmldata.nombres" type="text"/>
        <input label="Apellido parterno" name="okp:xmldata.apellido_padre" type="text
        <input label="Apellido materno" name="okp:xmldata.apellido_madre" type="text"
        <input label="Año contable" name="okp:xmldata.year" type="text"/>
        <input label="Tipo documento" name="okp:xmldata.documento" type="text"/>
    </property-group>
</property-groups>
```

XML files structure:

```
<?xml version="1.0" ?>
<INDEX_LOG>
    <Batch BatchID="2013-04-16">
        <File>
            <Filename>10201200365212 - 01.pdf</Filename>
            <Fields>
                <Field>
                    <Name>Folio</Name>
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
                    <Value>10201200365245</Value>
                </Field>
                <Field>
                    <Name>Rut</Name>
                    <Value>9419475-JK</Value>
                </Field>
                <Field>
                    <Name>Placa Patente</Name>
                    <Value>XG412190</Value>
                </Field>
                <Field>
                    <Name>Nombres</Name>
                    <Value>JOSEP</Value>
                </Field>
                <Field>
                    <Name>Apellido Paterno</Name>
                    <Value>LLORT</Value>
                </Field>
                <Field>
                    <Name>Apellido Materno</Name>
                    <Value>TELLA</Value>
                </Field>
                <Field>
                    <Name>Año</Name>
                    <Value>2012</Value>
                </Field>
                <Field>
                    <Name>Tipo de Documento</Name>
                    <Value>Comprobante de Permiso</Value>
                </Field>
            </Fields>
        </File>
        <!-- etc. -->    </Batch>
</INDEX_LOG>
```

JAVA class:

```
package com.openkm.plugin.cron;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

import com.openkm.api.OKMDocument;
import com.openkm.api.OKMPropertyGroup;
import com.openkm.api.OKMSearch;
import com.openkm.bean.Document;
import com.openkm.bean.PropertyGroup;
import com.openkm.bean.QueryResult;
```

```
import com.openkm.core.AccessDeniedException;
import com.openkm.core.DatabaseException;
import com.openkm.core.ItemExistsException;
import com.openkm.core.ParseException;
import com.openkm.core.PathNotFoundException;
import com.openkm.core.PromotedAsRecordLockException;
import com.openkm.core.RepositoryException;
import com.openkm.core.UnsupportedMimeTypeException;
import com.openkm.core.ValidationFormException;
import com.openkm.core.VirusDetectedException;
import com.openkm.db.bean.QueryParams;
import com.openkm.module.db.stuff.DbSessionManager;
import com.openkm.plugin.PluginNotFoundException;
import com.openkm.plugin.automation.AutomationException;
import com.openkm.principal.PrincipalAdapterException;

import net.xeoh.plugins.base.annotations.PluginImplementation;

@PluginImplementation
public class XmlImporter extends BaseCronPlugin implements CronAdapter {

    private static final String SYSTEM_FOLDER = "/home/openkm/import";
    private static final String OPENKM_FOLDER = "/okm:root/import";
    private static final String ATTRIBUTE_FILENAME = "Filename";
    private static final String ATTRIBUTE_FIELDS = "Fields";
    private static final String ATTRIBUTE_FIELD = "Field";
    private static final String ATTRIBUTE_NAME = "Name";
    private static final String ATTRIBUTE_VALUE = "Value";
    private static final String FIELD_NAME_FOLIO = "folio";
    private static final String FIELD_NAME_RUT = "rut";
    private static final String FIELD_NAME_PLACA_PATENTE = "placa patente";
    private static final String FIELD_NAME_NOMBRES = "nombres";
    private static final String FIELD_NAME_APELLIDO_PATERNO = "apellido paterno";
    private static final String FIELD_NAME_APELLIDO_MATERNO = "apellido materno";
    private static final String FIELD_NAME_ANO = "año";
    private static final String FIELD_NAME_TIPO_DOCUMENTO = "tipo de documento";

    private Logger log = LoggerFactory.getLogger(XmlImporter.class);

    @Autowired
    private OKMDocument okmDocument;

    @Autowired
    private OKMSearch okmSearch;

    @Autowired
    private OKMPropertyGroup okmPropertyGroup;

    private String token = DbSessionManager.getInstance().getSystemToken();

    @Override
    public String getName() {
        return "Xml Importer";
    }

    @Override
    public String getCronExpression() {
        return "0 5 * * * *";
    }

    @Override
    public void execute() {
        try {
            importFiles();
            importMetadata();
        } catch (Exception e) {
            log.error(e.getMessage());
```

```
            e.getStackTrace();
        }
    }

    /**
     * importMetadata
     */
    public void importMetadata() throws PathNotFoundException, AccessDeniedException,
            AutomationException, PluginNotFoundException, ValidationFormException, Pa
            ParserConfigurationException, SAXException {
        String fileName = "";
        String folio = "";
        String rut = "";
        String placaPatente = "";
        String nombres = "";
        String apellidoPaterno = "";
        String apellidoMaterno = "";
        String ano = "";
        String tipoDocumento = "";

        File folder = new File(SYSTEM_FOLDER + "/logfile");
        File[] listOfFiles = folder.listFiles();

        for (int i = 0; i < listOfFiles.length; i++) {
            File xmlFile = listOfFiles[i];
            if (xmlFile.isFile() && xmlFile.getName().toLowerCase().endsWith("indexlo
                DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
                dbf.setNamespaceAware(true);
                dbf.setAttribute("http://xml.org/sax/features/namespaces", Boolean.TR

                DocumentBuilder db = dbf.newDocumentBuilder();
                org.w3c.dom.Document xmlDoc = db.parse(xmlFile);
                xmlDoc.getDocumentElement().normalize();
                NodeList fileNodesList = xmlDoc.getElementsByTagName("File");

                for (int x = 0; x < fileNodesList.getLength(); x++) {
                    fileName = "";
                    folio = "";
                    rut = "";
                    placaPatente = "";
                    nombres = "";
                    apellidoPaterno = "";
                    apellidoMaterno = "";
                    ano = "";
                    tipoDocumento = "";
                    Node childFildNode = fileNodesList.item(x);

                    if (childFildNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element fileElement = (Element) childFildNode; // attibute no
                        fileName = fileElement.getElementsByTagName(ATTRIBUTE_FILENAM
                        Node fieldsNode = fileElement.getElementsByTagName(ATTRIBUTE_

                        // Fields
                        if (fieldsNode.getNodeType() == Node.ELEMENT_NODE) {
                            Element fieldsElement = (Element) fieldsNode; // fields

                            // node
                            NodeList fieldList = fieldsElement.getElementsByTagName(A
                            for (int y = 0; y < fieldList.getLength(); y++) {
                                Node fieldNode = fieldList.item(y);
                                if (fieldNode.getNodeType() == Node.ELEMENT_NODE) {
                                    Element fieldElement = (Element) fieldNode; // at

                                    // node
                                    String name = fieldElement.getElementsByTagName(A
                                            .trim();
                                    String value = fieldElement.getElementsByTagName(
```

```
                                        .trim();

                                if (name.toLowerCase().equals(FIELD_NAME_FOLIO))
                                    folio = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_R
                                    rut = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_P
                                    placaPatente = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_N
                                    nombres = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_A
                                    apellidoPaterno = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_A
                                    apellidoMaterno = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_A
                                    ano = value;
                                } else if (name.toLowerCase().equals(FIELD_NAME_T
                                    tipoDocumento = value;
                                }
                            }
                        }
                    }

                    if (fileName != null && !fileName.equals("")) {
                        QueryParams queryParams = new QueryParams();
                        queryParams.setDomain(QueryParams.DOCUMENT);
                        queryParams.setName(fileName);
                        List<QueryResult> results = okmSearch.find(token, queryPa

                        if (results.size() == 1) {
                            for (QueryResult queryResult : results) {
                                if (queryResult.getNode() != null && queryResult.
                                    log.info(fileName + " -ok");
                                    boolean found = false;

                                    for (PropertyGroup group : okmPropertyGroup.g
                                            queryResult.getNode().getPath())) {
                                        if (group.getName().equals("okg:xmldata")
                                            found = true;
                                        }
                                    }

                                    Map<String, String> properties = new HashMap<
                                    properties.put("okp:xmldata.folio", folio);
                                    properties.put("okp:xmldata.rut", rut);
                                    properties.put("okp:xmldata.placa", placaPate
                                    properties.put("okp:xmldata.nombres", nombres
                                    properties.put("okp:xmldata.apellido_padre",
                                    properties.put("okp:xmldata.apellido_madre",
                                    properties.put("okp:xmldata.year", ano);
                                    properties.put("okp:xmldata.documento", tipoD
                                    if (!found) {
                                        okmPropertyGroup.addGroup(token, queryRes
                                                properties);
                                    } else {
                                        okmPropertyGroup.setProperties(token, que
                                                properties);
                                    }
                                }
                            }
                        } else if (results.size() > 1) {
                            log.error(fileName + " - error");
                        } else {
                            log.error(fileName + " - not found");
                        }
                    }
                }
```

```
                }

                xmlFile.delete();
            }
        }
    }

    /**
     * importFiles
     */
    public void importFiles() throws UnsupportedMimeTypeException, VirusDetectedExcep
            PathNotFoundException, AccessDeniedException, RepositoryException, Databa
            PromotedAsRecordLockException, PrincipalAdapterException, IOException {
        // Loading files
        File folder = new File(SYSTEM_FOLDER + "/Output");
        File[] listOfFiles = folder.listFiles();
        for (int i = 0; i < listOfFiles.length; i++) {
            File file = listOfFiles[i];
            if (file.isFile() && file.getName().toLowerCase().endsWith(".pdf")) {
                Document doc = new Document();
                doc.setPath(OPENKM_FOLDER + "/" + file.getName());
                FileInputStream fis = new FileInputStream(file);
                doc = okmDocument.create(token, doc, fis);
                file.delete();
            }
        }
    }
}
```

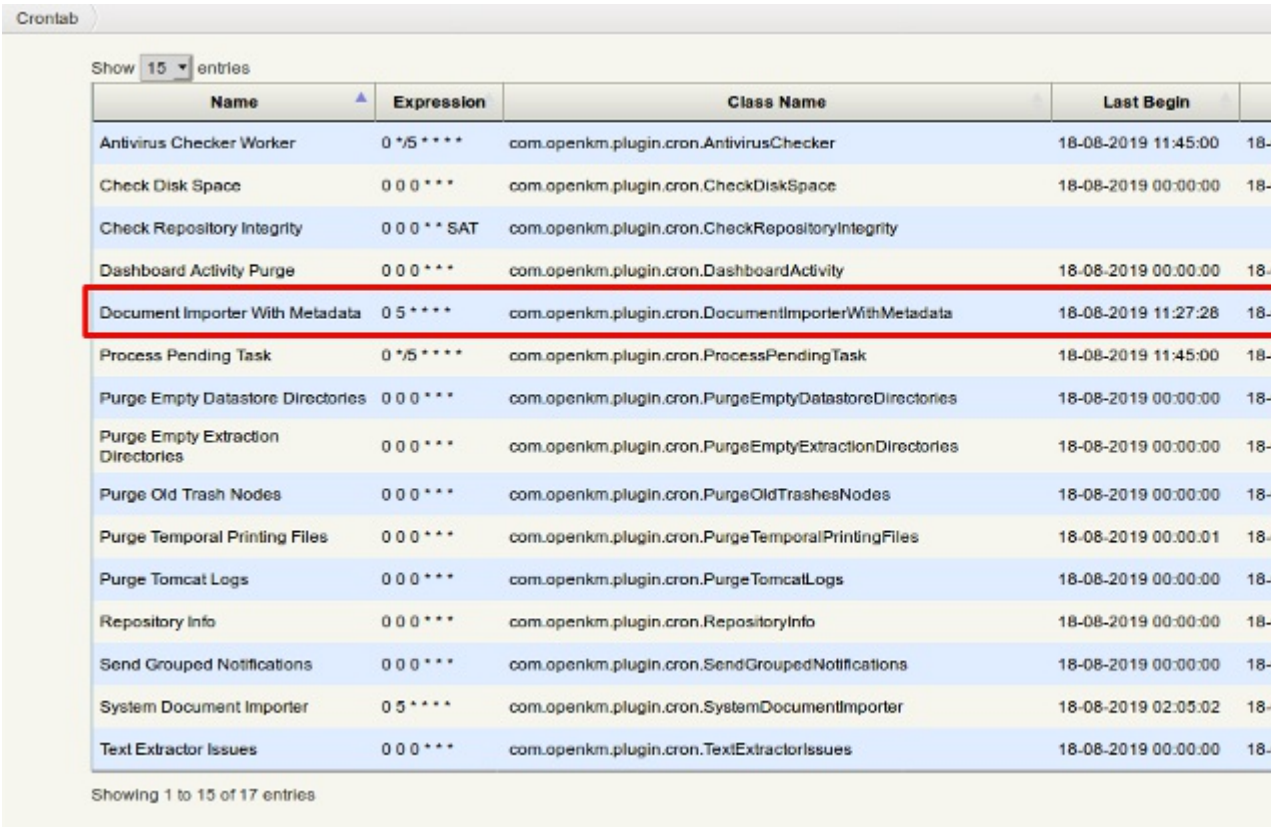
Register a new plugin

- *To install new plugin **create a jar file** and **copy into your $TOMCAT/plugins** folder. More information at Creating a JAR file with Eclipse.*
- *Go to **Administration > Utilities > Plugins** and click at top right the Reload plugins button.*
- *In **plugins table will be shown** the new plugin.*

Images

Crontab task

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


Crontab

Show 15 entries

Name	Expression	Class Name	Last Begin	
Antivirus Checker Worker	0 */5 * * * *	com.openkm.plugin.cron.AntivirusChecker	18-08-2019 11:45:00	18-
Check Disk Space	0 0 0 * * *	com.openkm.plugin.cron.CheckDiskSpace	18-08-2019 00:00:00	18-
Check Repository Integrity	0 0 0 * * SAT	com.openkm.plugin.cron.CheckRepositoryIntegrity		
Dashboard Activity Purge	0 0 0 * * *	com.openkm.plugin.cron.DashboardActivity	18-08-2019 00:00:00	18-
Document Importer With Metadata	0 5 * * * *	com.openkm.plugin.cron.DocumentImporterWithMetadata	18-08-2019 11:27:28	18-
Process Pending Task	0 */5 * * * *	com.openkm.plugin.cron.ProcessPendingTask	18-08-2019 11:45:00	18-
Purge Empty Datastore Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyDatastoreDirectories	18-08-2019 00:00:00	18-
Purge Empty Extraction Directories	0 0 0 * * *	com.openkm.plugin.cron.PurgeEmptyExtractionDirectories	18-08-2019 00:00:00	18-
Purge Old Trash Nodes	0 0 0 * * *	com.openkm.plugin.cron.PurgeOldTrashesNodes	18-08-2019 00:00:00	18-
Purge Temporal Printing Files	0 0 0 * * *	com.openkm.plugin.cron.PurgeTemporalPrintingFiles	18-08-2019 00:00:01	18-
Purge Tomcat Logs	0 0 0 * * *	com.openkm.plugin.cron.PurgeTomcatLogs	18-08-2019 00:00:00	18-
Repository Info	0 0 0 * * *	com.openkm.plugin.cron.RepositoryInfo	18-08-2019 00:00:00	18-
Send Grouped Notifications	0 0 0 * * *	com.openkm.plugin.cron.SendGroupedNotifications	18-08-2019 00:00:00	18-
System Document Importer	0 5 * * * *	com.openkm.plugin.cron.SystemDocumentImporter	18-08-2019 02:05:02	18-
Text Extractor Issues	0 0 0 * * *	com.openkm.plugin.cron.TextExtractorIssues	18-08-2019 00:00:00	18-

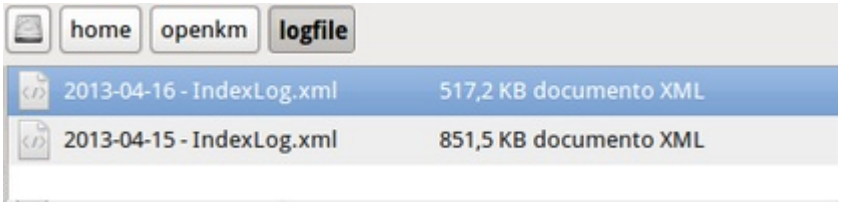
Showing 1 to 15 of 17 entries

Register metadata definition

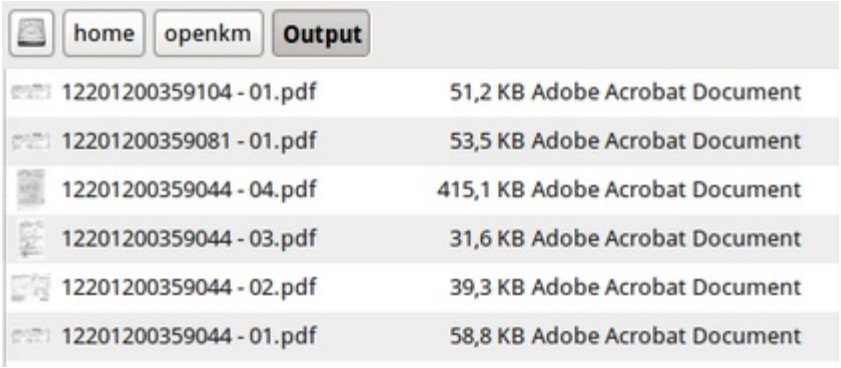
Metadata contracts

Group label		Group name			
Contract		okg:contract			
Label	**Name**	**Width**	**Height**	**Element**	
Cups	okp:contract.cups	200px	25px	Input	*Readonly:* false *Data:* *Type:* text *Value:* *Description:*
Año	okp:contract.year	200px	25px	Input	*Readonly:* false *Data:* *Type:* text *Value:* *Description:*
Mes	okp:contract.month	150px	25px	Select	*Readonly:* false *Data:* *Description:* *Type:* simple *Options:* • *Label:* Enero, *Value:* enero • *Label:* Febrero, *Value:* febr • *Label:* Marzo, *Value:* marzo • *Label:* Abril, *Value:* abril • *Label:* Mayo, *Value:* mayo • *Label:* Junio, *Value:* junio • *Label:* Julio, *Value:* julio • *Label:* Agosto, *Value:* agos • *Label:* Septiembre, *Value:* • *Label:* Octubre, *Value:* octu • *Label:* Noviembre, *Value:* • *Label:* Diciembre, *Value:* di

Logfile folder

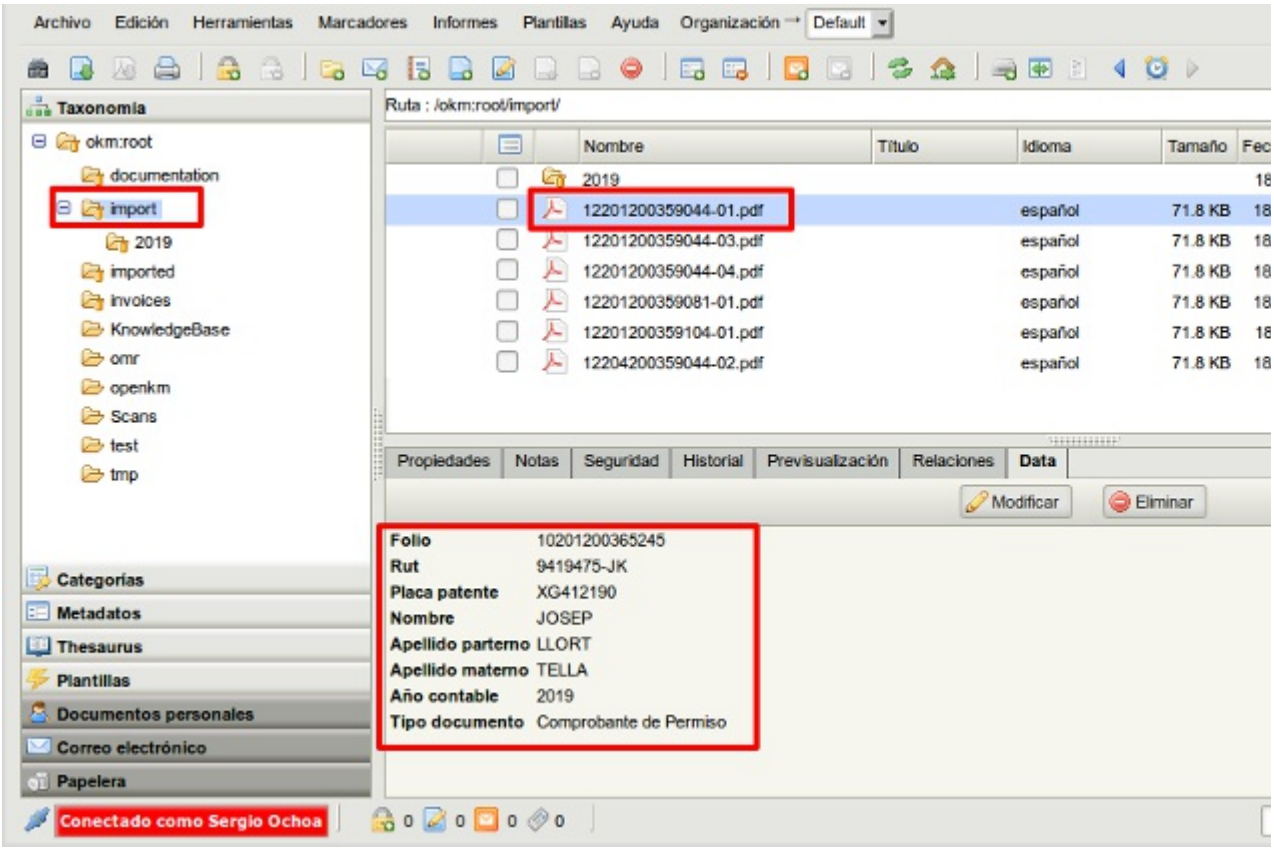


Output folder



Result

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.


Archivo
Edición
Herramientas
Marcadores
Informes
Plantillas
Ayuda
Organización
Default
Taxonomia
okm:root
documentation
import
2019
imported
invoices
KnowledgeBase
omr
openkm
Scans
test
tmp
Categorías
Metadatos
Thesaurus
Plantillas
Documentos personales
Correo electrónico
Papelera
Conectado como Sergio Ochoa
Ruta : /okm:root/import/
Nombre
Título
Idioma
Tamaño
2019
12201200359044-01.pdf
12201200359044-03.pdf
12201200359044-04.pdf
12201200359081-01.pdf
12201200359104-01.pdf
12204200359044-02.pdf
español
71.8 KB
Propiedades
Notas
Seguridad
Historial
Previsualización
Relaciones
Data
Modificar
Eliminar
Folio 10201200365245
Rut 9419475-JK
Placa patente XG412190
Nombre JOSEP
Apellido parterno LLORT
Apellido materno TELLA
Año contable 2019
Tipo documento Comprobante de Permiso

Variables by automation event

Events can be pre (previously) and post (posterior). For example ***pre*** *document creation event, will be executed before document creation and then can be changed several event vars to change parent folder, document name among others. In case* ***post*** *document creation event, will be executed after document creation. In this case there're no associated vars to the event you'll get NodeDocument object to apply directly any change across OpenKM API.*

Available environment variables

Variable	*Object class*	*Description*
AutomationUtils.TENANT	*long*	*Tenant*
AutomationUtils.CATEGORIES	*Set<String>*	*List of categories*
AutomationUtils.CREATION_DATE	*Calendar*	*Creation date*
AutomationUtils.DESTINATION_NODE	*NodeFolder*	*Destination folder*
AutomationUtils.DOCUMENT_NODE	*NodeDocument*	*Document node*
AutomationUtils.DOCUMENT_FILE	*File*	*Document File*
AutomationUtils.DOCUMENT_NAME	*String*	*Document name*
AutomationUtils.DOCUMENT_VERSION	*String*	*Document version*
AutomationUtils.EVENT	*AutomationRule.EnumEvents*	*Event*
AutomationUtils.EVENT_AT	*String*	*Event at*
AutomationUtils.FOLDER_NODE	*NodeFolder*	*Folder node*
AutomationUtils.KEYWORDS	*Set<String>*	*List of keywords*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.NODE_BASE	*NodeBase* • *NodeDocument* • *NodeFolder* • *NodeMail* • *NodeRecord* *These nodes extends from NodeBase.*	*Node base*
AutomationUtils.NODE_BASE_LIST	*List<NodeBase>* • *NodeDocument* • *NodeFolder* • *NodeMail* • *NodeRecord* *These nodes extends from NodeBase.*	*Node base list*
AutomationUtils.NOTES	*List<NodeNote>*	*List of notes*
AutomationUtils.MAIL_BCC	*String[]*	*Mail bcc*
AutomationUtils.MAIL_CC	*String[]*	*Mail cc*
AutomationUtils.MAIL_CONTENT	*String*	*Mail content*
AutomationUtils.MAIL_FROM	*String*	*Mail from*
AutomationUtils.MAIL_NODE	*NodeMail*	*Mail node*
AutomationUtils.MAIL_RECEIVED_DATE	*Calendar*	*Mail sent date*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.MAIL_REPLY	*String[]*	*Mail reply*
AutomationUtils.MAIL_SENT_DATE	*Calendar*	*Mail sent date*
AutomationUtils.MAIL_SUBJECT	*String*	*Mail subject*
AutomationUtils.MAIL_TO	*String[]*	*Mail to*
AutomationUtils.MIME_TYPE	*String*	*Mime Type*
AutomationUtils.NAME	*String*	*Name*
AutomationUtils.PROPERTY_GROUP_NAME	*String*	*Property group name*
AutomationUtils.PROPERTY_GROUP_PROPERTIES	*Map<String, String>*	*Property group properties*
AutomationUtils.PROPERTY_GROUPS	*Set<NodeProperty>*	*List of property groups (metadata groups)*
AutomationUtils.RECORD_NODE	*NodeRecord*	*Record node*
AutomationUtils.TEXT_EXTRACTED	*String*	*Document text extracted*
AutomationUtils.LANGUAGE_DETECTED	*String*	*Language detected*
AutomationUtils.UPLOAD_RESPONSE	*String*	*Upload response*
AutomationUtils.TITLE	*String*	*Title*
AutomationUtils.WIKIPAGE	*wikiPage*	*The wiki page*
AutomationUtils.NODE_CLASS	*Long*	*The NodeClass id*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.NEW_NODE_CLASS	*Long*	*The new NodeClass*
AutomationUtils.OLD_NODE_CLASS	*Long*	*The old NodeClass*
AutomationUtils.TASK_ID	*Long*	*The Task id*
AutomationUtils.TASK_OBJECT	*TaskManagerTask*	*Task object*
AutomationUtils.USER	*com.openkm.bean.CommonUser*	*User*
AutomationUtils.DOCUMENT_SIZE	*Long*	*Document size*

Document

Create Document

At Pre

Variable	***For changing***
AutomationUtils.CATEGORIES	*Yes*
AutomationUtils.CREATION_DATE	*No*
AutomationUtils.NAME	*Yes*
AutomationUtils.MIME_TYPE	*Yes*
AutomationUtils.KEYWORDS	*Yes*
AutomationUtils.NOTES	*Yes*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.NODE_CLASS ***At Pre***	*Yes*
AutomationUtils.DESTINATION_NODE	*Yes*
AutomationUtils.PROPERTY_GROUPS	*Yes*
AutomationUtils.TITLE	*Yes*
AutomationUtils.WIKIPAGE	*Yes*
AutomationUtils.DOCUMENT_SIZE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DOCUMENT_FILE	*No*

Update Document

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	

Delete Document

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Rename Document

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Purge Document

At Pre	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Variable	***For changing***
At Pre	
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.TENANT	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.TENANT	*No*

Move Document

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DESTINATION_NODE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Restore Version Document

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.DOCUMENT_VERSION	*No*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.DOCUMENT_VERSION	*No*

Download Document from UI

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DOCUMENT_VERSION	*No*
AutomationUtils.DOCUMENT_NAME	*Yes*
AutomationUtils.DOCUMENT_NODE	*No*
At post	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Variable	***For changing***
At Pre	
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DOCUMENT_VERSION	*No*
AutomationUtils.DOCUMENT_NAME	*No*
AutomationUtils.DOCUMENT_NODE	*No*

Download Document from UI for preview

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DOCUMENT_NAME	*Yes*
AutomationUtils.DOCUMENT_VERSION	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.DOCUMENT_NAME	*No*
AutomationUtils.DOCUMENT_VERSION	*No*

Print Document

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.DOCUMENT_NODE	
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.DOCUMENT_NODE	*No*

Stamp Document

At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Virus detected Document

At Pre	
Variable	***For changing***
AutomationUtils.USER	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.DOCUMENT_NAME ***At Pre***	*No*
AutomationUtils.VIRUS_INFO	*No*

Set Node Class Document

At Pre	
Variable	***For changing***
AutomationUtils.NODE_CLASS	*No*
AutomationUtils.OLD_NODE_CLASS	*No*
AutomationUtils.NEW_NODE_CLASS	*Yes*
AutomationUtils.DOCUMENT_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*
AutomationUtils.NODE_CLASS	*No*

Folder

Create Folder

At Pre	
Variable	***For changing***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.CATEGORIES ***At Pre***	*Yes*
AutomationUtils.CREATION_DATE	*No*
AutomationUtils.DESTINATION_NODE	*Yes*
AutomationUtils.NAME	*Yes*
AutomationUtils.NOTES	*Yes*
AutomationUtils.KEYWORDS	*Yes*
AutomationUtils.PROPERTY_GROUPS	*Yes*
AutomationUtils.WIKIPAGE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*

Delete Folder

At Pre	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*
At post	
Variable	***For changing***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
AutomationUtils.FOLDER_NODE	*No*

Move Folder

At Pre	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*
AutomationUtils.DESTINATION_NODE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*

Export Folder as ZIP

At Pre	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Export Folder as JAR

At Pre	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.FOLDER_NODE	*No*

Mail

Create mail

At Pre	
Variable	***For changing***
AutomationUtils.CATEGORIES	*Yes*
AutomationUtils.CREATION_DATE	*No*
AutomationUtils.DESTINATION_NODE	*Yes*
AutomationUtils.MAIL_FROM	*No*
AutomationUtils.MAIL_REPLY	*No*
AutomationUtils.MAIL_TO	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.MAIL_CC ***At Pre***	*No*
AutomationUtils.MAIL_BCC	*No*
AutomationUtils.MAIL_SENT_DATE	*No*
AutomationUtils.MAIL_RECEIVED_DATE	*No*
AutomationUtils.MAIL_SUBJECT	*No*
AutomationUtils.MAIL_CONTENT	*No*
AutomationUtils.NAME	*Yes*
AutomationUtils.KEYWORDS	*Yes*
AutomationUtils.MIME_TYPE	*No*
AutomationUtils.TITLE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*

Move Mail

At Pre	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
AutomationUtils.DESTINATION_NODE	*Yes*
At post	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*

Download mail from UI

At Pre	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*

Set Node Class Mail

At Pre	
Variable	***For changing***
AutomationUtils.NODE_CLASS	*No*
AutomationUtils.OLD_NODE_CLASS	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.NEW_NODE_CLASS ***At Pre***	*Yes*
AutomationUtils.MAIL_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*
AutomationUtils.NODE_CLASS	*No*

Mail import attachments Mail

At post	
Variable	***For changing***
AutomationUtils.MAIL_NODE	*No*

Record

Create record

At Pre	
Variable	***For changing***
AutomationUtils.DESTINATION_NODE	*Yes*
AutomationUtils.NAME	*Yes*
AutomationUtils.NODE_CLASS	*Yes*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.TITLE ***At Pre***	*Yes*
At post	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*

Delete record

At Pre	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*

Move Record

At Pre	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*
AutomationUtils.DESTINATION_NODE	*Yes*
At post	

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*

Set Node Class Record

At Pre	
Variable	***For changing***
AutomationUtils.NODE_CLASS	*No*
AutomationUtils.OLD_NODE_CLASS	*No*
AutomationUtils.NEW_NODE_CLASS	*Yes*
AutomationUtils.RECORD_NODE	*No*
At post	
Variable	***For changing***
AutomationUtils.RECORD_NODE	*No*
AutomationUtils.NODE_CLASS	*No*

Property Group (metadata)

Add property group

At Pre	
Variable	***For changing***

At Pre	
AutomationUtils.NODE_BASE	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*
AutomationUtils.PROPERTY_GROUP_PROPERTIES	*Yes*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*
AutomationUtils.PROPERTY_GROUP_PROPERTIES	*No*

Set property group

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*
AutomationUtils.PROPERTY_GROUP_PROPERTIES	*Yes*
At post	
Variable	***For changing***

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.NODE_BASE ***At Pre***	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*
AutomationUtils.PROPERTY_GROUP_PROPERTIES	*No*

Remove property group

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.PROPERTY_GROUP_NAME	*No*

Auth

Create user

At Pre	
Variable	***For changing***
AutomationUtils.USER	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At post *At Pre*	
Variable	***For changing***
AutomationUtils.USER	*No*

User login

At post	
Variable	***For changing***
AutomationUtils.USER	*No*

User logout

At Pre	
Variable	***For changing***
AutomationUtils.USER	*No*

Task

Create Task

At Pre	
Variable	***For changing***
AutomationUtils.TASK_OBJECT	*Yes*
At post	

Variable	***For changing***
At Pre	
AutomationUtils.TASK_OBJECT	*No*

Update Task

At Pre	
Variable	***For changing***
AutomationUtils.TASK_OBJECT	*Yes*
At post	
Variable	***For changing***
AutomationUtils.TASK_OBJECT	*No*

Delete Task

At Pre	
Variable	***For changing***
AutomationUtils.TASK_ID	*No*
At post	
Variable	***For changing***
AutomationUtils.TASK_ID	*No*

Note

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Create Note

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.NOTE_TEXT	*No*
At post	
Variable	***For changing***
AutomationUtils.NOTE_UUID	*No*

Update Note

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.NOTE_TEXT	*No*
AutomationUtils.NOTE_UUID	*No*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

At Pre	
AutomationUtils.NOTE_TEXT	*No*
AutomationUtils.NOTE_UUID	*No*

Delete Note

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.NOTE_UUID	*No*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE	*No*
AutomationUtils.NOTE_UUID	*No*

General

Text extraction

At Pre	
Variable	***For changing***
AutomationUtils.DOCUMENT_NODE	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

AutomationUtils.TENANT	*No*
At Pre	
At post	
Variable	***For changing***
AutomationUtils.TEXT_EXTRACTED	*Yes*
AutomationUtils.LANGUAGE_DETECTED	*Yes*

Export nodes to zip

At Pre	
Variable	***For changing***
AutomationUtils.NODE_BASE_LIST	*No*
At post	
Variable	***For changing***
AutomationUtils.NODE_BASE_LIST	*No*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Vault of stuff

Field / Property	*Description*
Convert Exchange Legacy DN	*Changes all the Legacy DN strings into proper mail addresses.*
Download document with direct link	*Downloads a document by url.*
Import document with bash script using REST	*Imports a document with a bash script that uses REST Webservices API.*
Creating users from scripts	*Creation of users from generated scripts with file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Convert Exchange Legacy DN

The script changes all the Legacy DN strings into proper mail addresses. Therefore there's an export file of the global address list from our IT as .csv containing all the Exchange Legacy DNs and their corresponding mail addresses. With this data the database can be altered the Exchange Legacy DN can be changed to normal mail addresses.

CSV sample:

```
/o=EMail/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=somefanc
/o=EMail/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=anotherf
/o=EMail/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=m.muelle
```

```
<?php
# Change ExchangeDN to Mailadresses
# Malte Küppers
#
# CSV Legacy File Format:        Exchange Legacy DN ; Mailadresse
#
#


# variables
$row = 0;
$legacy_file = "/root/Hilfsdateien/LegacyExchangeDN.txt" # Set to your filepath


if (($handle = fopen($legacy_file, "r")) !== FALSE) {
    while (($data = fgetcsv($handle, 10000, ";")) !== FALSE) {
                $exchdn[$row]=strtoupper($data[0]);     #Exchange Legacy DN
                $mail[$row]=$data[1];                           #Mailadresse
                $row++;

    }
    fclose($handle);
        #echo $row ." lines read";
}

#Open Database Connection
$db = mysqli_connect("127.0.0.1", "<USERNAME>", "<PASSWORD>", "okmdb");
if(!$db)
{
        exit("Verbindungsfehler: ".mysqli_connect_error());
}

# Mail FROM ändern
$ergebnis = mysqli_query($db, "SELECT DISTINCT NML_FROM FROM OKM_NODE_MAIL");
while($row = mysqli_fetch_object($ergebnis)){
                #only adresses that needs to be changed (Starting with /o= )
                if(strpos($row->NML_FROM,"/o")!==false) {

                        $key = array_search(strtoupper($row->NML_FROM), $exchdn);
                        if(is_numeric($key)){
                                echo $row->NML_FROM ." found at Pos #".$key ."<br>";
                                $qry = "UPDATE OKM_NODE_MAIL SET NML_FROM = '".$mail[
                                #echo $qry ."<br><br>";
                                $update = mysqli_query($db,$qry);
                                }
                }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
}

# Mail TO ändern
$ergebnis = mysqli_query($db, "SELECT DISTINCT NML_TO FROM OKM_NODE_MAIL_TO");
while($row = mysqli_fetch_object($ergebnis)){
                #Nur Adressen die nicht geändert wurden
                if(strpos($row->NML_TO,"/o")!==false) {

                        $key = array_search(strtoupper($row->NML_TO), $exchdn);
                        if(is_numeric($key)){
                                echo $row->NML_TO ." gefunden an Pos #".$key ."<br>";
                                $qry = "UPDATE OKM_NODE_MAIL_TO SET NML_TO = '".$mail
                                #echo $qry ."<br><br>";
                                $update = mysqli_query($db,$qry);
                                }
                }
}
# Mail CC ändern
$ergebnis = mysqli_query($db, "SELECT DISTINCT NML_CC FROM OKM_NODE_MAIL_CC");
while($row = mysqli_fetch_object($ergebnis)){
                #Nur Adressen die nicht geändert wurden
                if(strpos($row->NML_CC,"/o")!==false){

                        $key = array_search(strtoupper($row->NML_CC), $exchdn);
                        if(is_numeric($key)){
                                echo $row->NML_CC ." gefunden an Pos #".$key ."<br>";
                                $qry = "UPDATE OKM_NODE_MAIL_CC SET NML_CC = '".$mail
                                #echo $qry ."<br><br>";
                                $update = mysqli_query($db,$qry);
                                }
                }
}


# Mail BCC ändern
$ergebnis = mysqli_query($db, "SELECT DISTINCT NML_BCC FROM OKM_NODE_MAIL_BCC");
while($row = mysqli_fetch_object($ergebnis)){
                #Nur Adressen die nicht geändert wurden
                if(strpos($row->NML_BCC,"/o")!==false){

                        $key = array_search(strtoupper($row->NML_BCC), $exchdn);
                        if(is_numeric($key)){
                                echo $row->NML_BCC ." gefunden an Pos #".$key ."<br>"
                                $qry = "UPDATE OKM_NODE_MAIL_BCC SET NML_BCC = '".$ma
                                #echo $qry ."<br><br>";
                                $update = mysqli_query($db,$qry);
                                }
                }
}

?>
```

Additional information at our public forum.

Download document with direct link

You can download a document from OpenKM using a command line utility like ***wget****. There is a service with BASIC authentication for this task. You can reference a document by UUID using the* ***uuid*** *parameter. In the same URL you need to specify the user and password.*

> *When you share the user and password for downloading files is a good practice configure the user profile disallowing all the features. That will lock UI access to the user.*

Available parameters:

- *The parameter* ***uuid*** *is the document UUID*
- *The parameter* ***inline*** *set when document must be downloaded as attachment or shown inline. It's a boolean parameter, allowed values are true or false.*

> *When you use parameter path, it is a good practice to encode url to prevent problems with special characters.*

Sample:

```
http://user:password@@localhost:8080/openkm/Download?uuid=def34bcc-3291-4e11-9aa3-e86
```

This ease some integration tasks:

```
$ wget http://okmAdmin:admin@localhost:8080/openkm/Download?uuid=xxx-yyy-zzz
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Import document with bash script using REST

Simplified version of the script which make use of RESTful API.

```
OKM_SERVICE="http://localhost:8080/openkm"
OKM_USER="okmAdmin"
OKM_PASSWORD="admin"
OKM_UUID="754aefd5-c11c-4954-a94e-b062cf3fed97"
OKM_PATH="/okm:root/upload"
LOCAL_PATH="."
TMP=$(mktemp)
RESET='\e[0m'
ERROR='\e[1;31m'
OK='\e[1;34m'

for DIR in $(find $LOCAL_PATH -type d); do
  DIR=${DIR:1:${#DIR}}
  if [ -n "$DIR" ]; then
    RETCODE=$(curl -u $OKM_USER:$OKM_PASSWORD --write-out %{http_code} -X POST --sile
      -H "Content-Type: application/json" -d "$DIR" "$OKM_SERVICE/rest/folders/parent
    if [ $RETCODE != "200" ]; then
      echo -e "${ERROR}ERROR -> $(cat $TMP)${RESET}";
    else
      echo -e "${OK}Folder created: ${OKM_PATH}${DIR}${RESET}"
    fi
  fi
done

for FILE in $(find $LOCAL_PATH -type f); do
  FILE=${FILE:2:${#FILE}}
  RETCODE=$(curl -u $OKM_USER:$OKM_PASSWORD --write-out %{http_code} -X POST --silent
    -F "docPath=$OKM_PATH/$FILE" -F content=@$FILE "$OKM_SERVICE/rest/documents/paren
  if [ $RETCODE != "200" ]; then
    echo -e "${ERROR}ERROR -> $(cat $TMP)${RESET}";
  else
    echo -e "${OK}Document created: ${OKM_PATH}/${FILE}${RESET}"
  fi
done

# Cleanup
rm -f $TMP
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Creating users from scripts

The script allows you to create users and assign them the user role.

Download file create-users_okm.xlsx

> *The string "**newId**" is the **user Id** what will be set.*
>
> *The string "**newPassword**" is the **user password** what will be set.*
>
> *The string "**newMail@mail.com**" is the **user mail** what will be set.*
>
> *The string "**newRealName**" is the **user real name** what will be set.*

```
import com.openkm.util.ContextWrapper;
import com.openkm.api.OKMAuth;
import com.openkm.bean.User;

OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
User user = new User("newId", "newPassword", "newMail@mail.com", "New Real Name", tru
okmAuth.createUser(null, user);
okmAuth.assignRole(null, user.getId(), "ROLE_USER");
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

API description

OpenKM API classes:

OpenKM API class	*Description*
OKMActivity	*Manage the activites and actions*
OKMAuth	*Manage security and users. For example add or remove grants on a node, create or modify users or getting the profiles.*
OKMBookmark	*Manage the user bookmarks.*
OKMDashboard	*Manage all data shown at dashboard.*
OKMDocument	*Manage documents. For example create, move or delete a document.*
OKMFolder	*Manage folders. For example create, move or delete a folder.*
OKMLink	*Manage links. For example create a link.*
OKMMail	*Manage mails. For example create, move or delete a mails.*
OKMNote	*Manage notes on any node type. For example create, edit or delete a note on a document, folder, mail or record.*
OKMNotification	*Manage notifications. For example add or remove subscriptions on a document or a folder.*
OKMProperty	*Manage categories and keywords. For example add or remove keywords on a document, folder, mail or record.*
OKMPropertyGroup	*Manage metadata. For example add metadata group, set metadata fields.*
OKMRecord	*Manage records. For example create, move or delete a record.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMRelation	*Manage relations between nodes. For example create a relation (parent-child) between two documents.*
OKMReport	*Manage reports.*
OKMRepository	*A lot of stuff related with repository. For example get the properties of main root node (/okm:root).*
OKMSearch	*Manage search feature. For example manage saved queries or perform a new query to the repository.*
OKMStats	*General stats of the repository.*
OKMTask	*Manage task. For example create a new task.*
OKMUserConfig	*Manage user home configuration.*
OKMWorkflow	*Manage workflows. For example execute a new workflow.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMActivity

With OKMActivity you can explore the OpenKM activity log data.

Methods

findByActivity

Description:

Method	*Return values*	*Description*
findByActivity(String token, ActivityFilter filter)	***List<Activity>***	*Returns a list of all the activity log.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMActivity;
import com.openkm.db.bean.Activity;
import com.openkm.db.bean.ActivityFilter;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMActivity okmActivity = ContextWrapper.getContext().getBean(OKMActivity

            Calendar beginDate = Calendar.getInstance();
            beginDate.add(Calendar.MONTH, -1);
            Calendar endDate = Calendar.getInstance();
            String item = "f84a2e1f-a858-4e53-9c09-36519d903782";
            String user = "okmAdmin";

            ActivityFilter filter = new ActivityFilter();
            filter.setBegin(beginDate);
            filter.setEnd(endDate);
            filter.setItem(item);
            filter.setUser(user);
            filter.setPage(0);
            filter.setLength(20);
            for (Activity activity : okmActivity.findByFilter(null, filter)) {
                System.out.println(activity);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countByFilter

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
countByFilter(String token, ActivityFilter filter)	***int***	*Returns the number of activity log rows.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMActivity;
import com.openkm.db.bean.ActivityFilter;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMActivity okmActivity = ContextWrapper.getContext().getBean(OKMActivity

            Calendar beginDate = Calendar.getInstance();
            beginDate.add(Calendar.MONTH, -1);
            Calendar endDate = Calendar.getInstance();
            String item = "f84a2e1f-a858-4e53-9c09-36519d903782";
            String user = "okmAdmin";

            ActivityFilter filter = new ActivityFilter();
            filter.setBegin(beginDate);
            filter.setEnd(endDate);
            filter.setItem(item);
            filter.setUser(user);
            filter.setPage(0);
            filter.setLength(20);
            System.out.println(okmActivity.countByFilter(null, filter));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getActions

Description:

Method	*Return values*	*Description*
getActions(String token)	***List<String>***	*Returns a list of all the activity log actions.*

Example

```
package com.openkm;

import com.openkm.api.OKMActivity;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMActivity okmActivity = ContextWrapper.getContext().getBean(OKMActivity

            for (String action : okmActivity.getActions(null)) {
                System.out.println(action);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMAuth

Used for managing security and users. For example add or remove grants on a node, create or modify users or getting the profiles.

Basics

*The class **com.openkm.bean.Permission** contains permission values (READ, WRITE, etc.). You should use it in combination with methods that are changing or getting security grants.*

To set READ and WRITE access you should do:

```
int permission = Permission.READ + Permission.WRITE;
```

To check if you have permission access you should do:

```
// permission is a valid integer value
if ((permission | Permission.WRITE) = Permission.WRITE) {
  // Has WRITE grants.
}
```

*On almost methods you'll see parameter named "**nodeId**". The value of this parameter can be some valid node **UUID** (folder, document, mail, record).*

Example of nodeId:

- *Using UUID -> "**b153c4b7-3d1c-4589-bd42-0ed0f34fd338**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

login

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
login()	**void**	*Simulates user UI login process.*

When user is logged from UI, are executed some background process what creates main user nodes, like /okm:trash. Unfortunatelly if the user has never logged from UI and login from API these nodes are still not created and will raise an error, for it is necessary at the beginning execute login method.

The user must be logged before executing the method.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.login();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

login

Description:

Method	Return values	Description
login(String user, String pass)	**String**	*Login process return authentication token.*

Login token by default have an expiration time of 24h.

After executing the login method, all the next calls will use automatically the authentication token.

Each time login method is executed the login token is replaced by newer.

When user login into the application the first time, is called internally a method what creates user specific folders, like /okm:trash/userId etc.

From API point of view, this method should be only executed first time user accessing from API, for creating

specific user folder structure.

Otherwise you can get errors, for example PathNotExistsException /okm:trash/userId when deleting objects, because the folders has not been created.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.login("userId","password");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

grantUser

Description:

Method	*Return values*	*Description*
grantUser(String token, String nodeId, String user, int permissions, boolean recursive)	***void***	*Add user grant on a node.*
The parameter recursive only has sense when the nodeId is a folder or record node. *When parameter recursive is true, the change will be applied to the node and descendants.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            // Add sochoa write grants at the node but not descendants
```

```
            okmAuth.grantUser(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "sochoa",

            // Add all okmAdmin grants at the node and descendants
            okmAuth.grantUser(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "okmAdmin
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

revokeUser

Description:

Method	*Return values*	*Description*
revokeUser(String token, String nodeId, String user, int permissions, boolean recursive)	***void***	*Remove user grant on a node.*
The parameter recursive only has sense when the nodeId is a folder or a record node. *When parameter recursive is true, the change will be applied to the node and descendants.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

            // Add sochoa write grants at the node but not descendants
            okmAuth.revokeUser(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "sochoa"

            // Remove all okmAdmin grants at the node and descendants
            okmAuth.revokeUser(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "okmAdmi
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getGrantedUsers

Description:

Method	*Return values*	*Description*

getGrantedUsers(String token, String nodeId)	***Map<String, Integer>***	*Return the granted users of a node.*

Example:

```
package com.openkm;

import java.util.Map;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

            Map<String, Integer> grants = okmAuth.getGrantedUsers(null, "b153c4b7-3d1
            for (String role : grants.keySet()) {
                System.out.println(role + "->" + grants.get(role));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

grantRole

Description:

Method	*Return values*	*Description*
grantRole(String token, String nodeId, String role, int permissions, boolean recursive)	***void***	*Add role grant on a node.*
The parameter recursive only has sense when the nodeId is a folder or a record node. *When parameter recursive is true, the change will be applied to the node and descendants.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

            // Add ROLE_USER write grants at the node but not descendants
            okmAuth.grantRole(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "ROLE_USE
            // Add all ROLE_ADMIN grants to the node and descendants
            okmAuth.grantRole(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "ROLE_ADM
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

revokeRole

Description:

Method	*Return values*	*Description*
revokeRole(String token, String nodeId, String role, int permissions, boolean recursive)	***void***	*Remove role grant on a node.*
The parameter recursive only has sense when the nodeId is a folder or a record node. *When parameter recursive is true, the change will be applied to the node and its descendants.*		

Example:

```
package com.openkm;
import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            // Remove ROLE_USER write grants at the node but not descendants
            okmAuth.revokeRole(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "ROLE_US

            // Remove all ROLE_ADMIN grants to the node and descendants
            okmAuth.revokeRole(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "ROLE_AD
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getGrantedRoles

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
getGrantedRoles(String token, String nodeId)	***Map<String, Integer>***	*Return the granted roles of a node.*

Example:

```
package com.openkm;

import java.util.Map;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            Map<String, Integer> grants = okmAuth.getGrantedRoles(null, "b153c4b7-3d1
            for (String role : grants.keySet()) {
                System.out.println(role + "->" + grants.get(role));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUsers

Description:

Method	*Return values*	*Description*
getUsers(String token)	***List<CommonUser>***	*Return the list of all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            for (CommonUser user : okmAuth.getUsers(null)) {
                System.out.println(user.getId());
                System.out.println(user.getName());
                System.out.println(user.getEmail());
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
                System.out.println(user.getActive());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRoles

Description:

Method	*Return values*	*Description*
getRoles(String token)	***List<String>***	*Return the list of all the roles.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            for (String role : okmAuth.getRoles(null)) {
                System.out.println(role);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUsersByRole

Description:

Method	*Return values*	*Description*
getUsersByRole(String token, String role)	***List<CommonUser>***	*Return the list of all the users who have assigned a role.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            for (CommonUser user : okmAuth.getUsersByRole(null, "ROLE_ADMIN")) {
                System.out.println(user.getId());
                System.out.println(user.getName());
                System.out.println(user.getEmail());
                System.out.println(user.getActive());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRolesByUser

Description:

Method	*Return values*	*Description*
getRolesByUser(String token, String user)	***List<String>***	*Return the list of all the roles assigned to a user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            for (String role : okmAuth.getRolesByUser(null, "okmAdmin")) {
                System.out.println(role);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

changeSecurity

Description:

Method	*Return values*	*Description*
public void changeSecurity(String token, String nodeId, Map<String, Integer>		

grantUsers, Map<String, Integer> revokeUsers, Map<String, Integer> grantRoles, Map<String, Integer> revokeRoles, boolean recursive)	***void***	*Change the security of a node.*

Example:

```
package com.openkm;

import java.util.HashMap;
import java.util.Map;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            String nodeId = "b153c4b7-3d1c-4589-bd42-0ed0f34fd338";
            Map<String, Integer> grantUsers = new HashMap<>();
            Map<String, Integer> revokeUsers = new HashMap<>();
            Map<String, Integer> grantRoles = new HashMap<>();
            grantRoles.put("ROLE_TEST", Permission.READ | Permission.WRITE);
            Map<String, Integer> revokeRoles = new HashMap<>();
            okmAuth.changeSecurity(null, nodeId, grantUsers, revokeUsers, grantRoles,
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createUser

Description:

Method	*Return values*	*Description*
createUser(String token, User user)	***void***	*Create a new user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.User;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            User user = new User();
```

```
            user.setId("test");
            user.setPassword("password.2019");
            user.setEmail("some@mail.com");
            user.setName("User Name");
            user.setActive(true);
            okmAuth.createUser(null, user);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteUser

Description:

Method	*Return values*	*Description*
deleteUser(String token, String user)	***void***	*Delete a user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.deleteUser(null, "test");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateUser

Description:

Method	*Return values*	*Description*
updateUser(String token, User user)	***void***	*Update a user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.User;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            User user = new User();
            user.setId("test");
            user.setPassword("newpassword");
            user.setEmail("some@mail.com");
            user.setName("Test");
            user.setActive(false);
            okmAuth.updateUser(null, user);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createRole

Description:

Method	*Return values*	*Description*
createRole(String token, Role role)	***void***	*Create a new role.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Role;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            Role role = new Role();
            role.setId("ROLE_TEST");
            role.setActive(true);
            okmAuth.createRole(null, role);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteRole

Description:

Method	*Return values*	*Description*

deleteRole(String token, String role)	***void***	*Delete a role.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.deleteRole(null, "ROLE_TEST");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateRole

Description:

Method	*Return values*	*Description*
updateRole(String token, String role, boolean active)	***void***	*Update a role.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;

public class Test {
    public static void main(String[] args) {
        try {
            OKMAuth.getInstance().updateRole(null, "ROLE_TEST",true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

assignRole

Description:

Method	*Return values*	*Description*
assignRole(String token, String user, String role)	***void***	*Assign role to a user.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.assignRole(null, "test", "ROLE_USER");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

removeRole

Description:

Method	*Return values*	*Description*
removeRole(String token, String user, String role)	***void***	*Remove a role from a user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.removeRole(null, "test", "ROLE_USER");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProfiles

Description:

Method	*Return values*	*Description*
getProfiles(String token, boolean filterByActive)	***List<Profile>***	*Return the list of all profiles.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The parameter filterByActive when enabled the method will return only the active profiles, otherwise will return all available profiles.

Each user has assigned one profile that enables more or less of the OpenKM UI features.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.db.bean.Profile;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            for (Profile profile : okmAuth.getProfiles(null, true)) {
                System.out.println(profile);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserProfile

Description:

Method	*Return values*	*Description*
getUserProfile(String token, String userId)	***Profile***	*Return the profile assigned to a user.*

Each user has assigned one profile that enables more or less of the OpenKM UI features.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
```

```
            System.out.println(okmAuth.getUserProfile(null, "okmAdmin"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setUserProfile

Description:

Method	*Return values*	*Description*
setUserProfile(String token, String userId, long profileId)	***void***	*Change the assigned profile to a user.*

Each user has assigned one profile that enables more or less of the OpenKM UI features.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.db.bean.Profile;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            // Set the profile named "default" to the user
            for (Profile profile : okmAuth.getProfiles(null, true)) {
                if (profile.getName().equals("default")) {
                    okmAuth.setUserProfile(null, "okmAdmin", profile.getId());
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

logout

Description:

Method	*Return values*	*Description*
logout()	*void*	*Execute logout method in OpenKM side.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            okmAuth.logout(null);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUser

Description:

Method	*Return values*	*Description*
getUser(String token, String userId)	***CommonUser***	*Return user data.*

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.CommonUser;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            CommonUser commonUser = okmAuth.getUser(null, "okmAdmin");
            System.out.print(commonUser);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setUserPermissions

Description:

Method	*Return values*	*Description*

public void setUserPermissions(String token, String nodeId, String user, int permissions, boolean recursive)	***void***	*Update user permissions on a node.*
The parameter recursive only has sense when the uuid is a folder or record node. *When parameter recursive is true, the changes will be applied to the node and descendants.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            // Update permissions of sochoa at the node but not descendants
            okmAuth.setUserPermissions(null, "3c68b3a1-c65c-4b1e-84b5-9ce2712ca573",

            // Update permissions of okmAdmin at the node and descendants
            okmAuth.setUserPermissions(null, "3c68b3a1-c65c-4b1e-84b5-9ce2712ca573",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setRolePermissions

Description:

Method	*Return values*	*Description*
setRolePermissions(String token, String nodeId, String role, int permissions, boolean recursive)	***void***	*Update role permissions on a node.*
The parameter recursive only has sense when the uuid is a folder or record node. *When parameter recursive is true, the changes will be applied to the node and descendants.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
```

```
import com.openkm.bean.Permission;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);
            // Update permissions of ROLE_USER at the node but not descendants
            okmAuth.setRolePermissions(null, "3c68b3a1-c65c-4b1e-84b5-9ce2712ca573",

            // Update permissions of ROLE_ADMIN at the node and descendants
            okmAuth.setRolePermissions(null, "3c68b3a1-c65c-4b1e-84b5-9ce2712ca573",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserTenants

Description:

Method	*Return values*	*Description*
getUserTenants(String token)	***List<Tenant>***	*Return the list of all tenants to which the user have access.*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMAuth;
import com.openkm.db.bean.Tenant;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

            List<Tenant> tenants = okmAuth.getUserTenants(null);
            for (Tenant tenant : tenants) {
                System.out.println(tenant);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setUserTenant

Description:

Method	*Return values*	*Description*

setUserTenant(String token, long tenantId)	***void***	*Change the assigned tenant to a user.*

User can have access to several tenants but only have one assigned.

Example:

```
package com.openkm;

import com.openkm.api.OKMAuth;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMAuth okmAuth = ContextWrapper.getContext().getBean(OKMAuth.class);

            long tenantId = 1;
            okmAuth.setUserTenant(null, tenantId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

OKMBookmark

Used for managing bookmarks of users. For example add or remove bookmark of a node for a user.

Basics

*On almost methods you'll see parameter named "**nodeId**". The value of this parameter can be some valid node **UUID** (folder, document, mail, record).*

Example of nodeId:

- *Using UUID -> "**3887ae75-dd67-4019-b4d1-2f6962815403**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

add

Description:

Method	*Return values*	*Description*
add(String token, String nodeId, String name)	***Bookmark***	*Add a new bookmark of a node.*
The parameter name indicates the name which will be used to store the bookmark. Usually is used the same name of the nodeId.		

Example:

```
package com.openkm;

import com.openkm.api.OKMBookmark;
import com.openkm.db.bean.Bookmark;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMBookmark okmBookmark = ContextWrapper.getContext().getBean(OKMBookmark
            Bookmark bookmark = okmBookmark.add(null, "3887ae75-dd67-4019-b4d1-2f6962
            System.out.println(bookmark);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

get

Description:

Method	*Return values*	*Description*
get(String token, int bmId)	***Bookmark***	*Gets the bookmark data.*
The parameter bmId is the value of the unique bookmark identifier.		

Example:

```
package com.openkm;

import com.openkm.api.OKMBookmark;
import com.openkm.db.bean.Bookmark;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMBookmark okmBookmark = ContextWrapper.getContext().getBean(OKMBookmark
            int bmId = 2;
            Bookmark bookmark = okmBookmark.get(null, bmId);
            System.out.println(bookmark);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

remove

Description:

Method	*Return values*	*Description*
remove(String token, int bmId)	***void***	*Delete a bookmark.*
The parameter bmId is the value of the unique bookmark identifier.		

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMBookmark;

public class Test {
    public static void main(String[] args) {
        try {
            int bmId = 23;
            OKMBookmark.getInstance().remove(null, bmId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

rename

Description:

Method	*Return values*	*Description*
rename(String token, int bmId, String newName	***Bookmark***	*Rename a bookmark.*
The parameter bmId is the value of the unique bookmark identifier.		

Example:

```
package com.openkm;

import com.openkm.api.OKMBookmark;
import com.openkm.db.bean.Bookmark;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMBookmark okmBookmark = ContextWrapper.getContext().getBean(OKMBookmark
            int bmId = 2;
            Bookmark bookmark = okmBookmark.rename(null, bmId, "newname.doc");
            System.out.println(bookmark);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAll

Description:

Method	*Return values*	*Description*

getAll(String token)	***List<Bookmark>***	*Gets all the bookmarks of the logged user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMBookmark;
import com.openkm.db.bean.Bookmark;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMBookmark okmBookmark = ContextWrapper.getContext().getBean(OKMBookmark
            for (Bookmark bookmark : okmBookmark.getAll(null)) {
                System.out.println(bookmark);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMDashboard

Basics

*On all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

getUserCheckedOutDocuments

Description:

Method	*Return values*	*Description*
getUserCheckedOutDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of documents checkout by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastModifiedDocuments

Description:

Method	Return values	Description
getUserLastModifiedDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the last documents modified by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLockedDocuments

Description:

Method	Return values	Description
getUserLockedDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the locked documents by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

getUserLockedRecords

Description:

Method	*Return values*	*Description*
getUserLockedRecords(String token, String user)	***List<DashboardRecordResult>***	*Get the list of the locked records by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardRecordResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardRecordResult dashboardRecordResult : okmDashboard.getUserLo
                System.out.println(dashboardRecordResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserSubscribedDocuments

Description:

Method	*Return values*	*Description*
getUserSubscribedDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the subscribed documents by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserSubscribedFolders

Description:

Method	*Return values*	*Description*
getUserSubscribedFolders(String token, String user)	***List<DashboardFolderResult>***	*Get the list of the subscribed folders by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardFolderResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardFolderResult dashboardFolderResult : okmDashboard.getUserSu
                System.out.println(dashboardFolderResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserSubscribedRecords

Description:

Method	*Return values*	*Description*
getUserSubscribedRecords(String token,	***List<DashboardRecordResult>***	*Get the list of the subscribed*

String user)		*records by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardRecordResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardRecordResult dashboardRecordResult : okmDashboard.getUserSu
                System.out.println(dashboardRecordResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastCreatedDocuments

Description:

Method	*Return values*	*Description*
getUserLastCreatedDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the last created documents by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getUserLastCreatedFolders

Description:

Method	*Return values*	*Description*
getUserLastCreatedFolders(String token, String user)	***List<DashboardFolderResult>***	*Get the list of the last created folders by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardFolderResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardFolderResult dashboardFolderResult : okmDashboard.getUserLa
                System.out.println(dashboardFolderResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastCreatedRecords

Description:

Method	*Return values*	*Description*
getUserLastCreatedRecords(String token, String user)	***List<DashboardRecordResult>***	*Get the list of the last created records by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardRecordResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
```

```
            for (DashboardRecordResult dashboardRecordResult : okmDashboard.getUserLa
                System.out.println(dashboardRecordResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastDownloadedDocuments

Description:

Method	*Return values*	*Description*
getUserLastDownloadedDocuments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the last download documents by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastImportedMails

Description:

Method	*Return values*	*Description*
getUserLastImportedMails(String token, String user)	***List<DashboardMailResult>***	*Get the list of the last imported mails by the user.*

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardMailResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardMailResult dashboardMailResult : okmDashboard.getUserLastIm
                System.out.println(dashboardMailResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserLastImportedMailAttachments

Description:

Method	*Return values*	*Description*
getUserLastImportedMailAttachments(String token, String user)	***List<DashboardDocumentResult>***	*Get the list of the last imported mail attachments by the user.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getUs
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserDocumentsSize

Description:

Method	*Return values*	*Description*

getUserDocumentsSize(String token)	***long***	*Return the total size in bytes of the user documents.*

> *When user upload or modify a document, the entire size of the contents - and history versions in case of update - are assigned to the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FormatUtil;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            long sizeInBytes = okmDashboard.getUserDocumentsSize(null);
            System.out.println(FormatUtil.formatSize(sizeInBytes));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUserSearchs

Description:

Method	*Return values*	*Description*
getUserSearchs(String token)	***List<QueryParams>***	*Get the list of all "user news" saved by the user.*

> *The "user saved" searches are executed periodically for getting news based on queries from the repository.*
>
> *More information at Working with the dashboard and Working with search views.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (QueryParams qp : okmDashboard.getUserSearchs(null)) {
                System.out.println(qp);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

find

Description:

Method	*Return values*	*Description*
find(String token, int qpId)	***List<DashboardNodeResult>***	*Execute a "user news".*
The param qpId is the unique identifier of the "users news" search.		

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardNodeResult;
import com.openkm.bean.Document;
import com.openkm.bean.Folder;
import com.openkm.bean.Mail;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            int qpId = 12;
            for (DashboardNodeResult dashboardNodeResult : okmDashboard.find(null, qp
                if (dashboardNodeResult.getNode() instanceof Document) {
                    Document doc = (Document) dashboardNodeResult.getNode();
                    System.out.println(doc);
                } else if (dashboardNodeResult.getNode() instanceof Folder) {
                    Folder fld = (Folder) dashboardNodeResult.getNode();
                    System.out.println(fld);
                } else if (dashboardNodeResult.getNode() instanceof Record) {
                    Record rec = (Record) dashboardNodeResult.getNode();
                    System.out.println(rec);
                } else if (dashboardNodeResult.getNode() instanceof Mail) {
                    Mail mail = (Mail) dashboardNodeResult.getNode();
                    System.out.println(mail);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
    }
}
```

getLastWeekTopDownloadedDocuments

Description:

Method	*Return values*	*Description*
getLastWeekTopDownloadedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the top downloaded documents during the last week by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastMonthTopDownloadedDocuments

Description:

Method	*Return values*	*Description*
getLastMonthTopDownloadedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the top downloaded documents during the last month by all the users.*

Example:

```
package com.openkm;
```

```
import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastWeekTopModifiedDocuments

Description:

Method	*Return values*	*Description*
getLastWeekTopModifiedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the top modified documents during the last week by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastMonthTopModifiedDocuments

Description:

Method	*Return values*	*Description*

getLastMonthTopModifiedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the top modified documents during the last month by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastModifiedDocuments

Description:

Method	*Return values*	*Description*
getLastModifiedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the last modified documents by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

getLastCreatedDocuments

Description:

Method	*Return values*	*Description*
getLastCreatedDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the last created documents by all the users.*

Example:

```
package com.openkm;
import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastCheckedOutDocuments

Description:

Method	*Return values*	*Description*
getLastCheckedOutDocuments(String token)	***List<DashboardDocumentResult>***	*Get the list of the last documents on edition by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardDocumentResult;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardDocumentResult dashboardDocumentResult : okmDashboard.getLa
                System.out.println(dashboardDocumentResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastCreatedFolders

Description:

Method	*Return values*	*Description*
getLastCreatedFolders(String token)	***List<DashboardFolderResult>***	*Get the list of the last created folders by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardFolderResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardFolderResult dashboardFolderResult : okmDashboard.getLastCr
                System.out.println(dashboardFolderResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLastCreatedRecords

Description:

Method	*Return values*	*Description*
getLastCreatedRecords(String token)	***List<DashboardRecordResult>***	*Get the list of the last created records by all the users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDashboard;
import com.openkm.bean.DashboardRecordResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashbo
            for (DashboardRecordResult dashboardRecordResult : okmDashboard.getLastCr
                System.out.println(dashboardRecordResult);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

visiteNode

Description:

Method	*Return values*	*Description*
visiteNode(String token, String source, String node, Calendar date)	***void***	*Mark a node as a visited.*

> *From UI, the nodes are shown strong until are visited by the user. This method mark the node as visited by the user in the database.*

Available sources:

- *LastWeekTopDownloadedDocuments*
- *LastMonthTopDownloadedDocuments*
- *LastWeekTopModifiedDocuments*
- *LastMonthTopModifiedDocuments*
- *LastModifiedDocuments*
- *LastCreatedDocuments*
- *LastCreatedFolders*
- *LastCreatedRecords*
- *LastCheckoutDocuments*
- *UserLockedDocuments*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *UserLockedRecord*
- *UserCheckedOutDocuments*
- *UserLastModifiedDocuments*
- *UserLastDownloadedDocuments*
- *UserSubscribedDocuments*
- *UserSubscribedFolders*
- *UserSubscribedRecords*
- *UserLastCreatedDocuments*
- *UserLastCreatedFolders*
- *UserLastCreatedRecords*
- *UserLastImportedMails*
- *UserLastImportedMailAttachments*
- *In case "user new", the source name is the name of the query.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMDashboard;
import com.openkm.util.ContextWrapper;

public class Test {

 public static void main(String[] args) {
  try {
   OKMDashboard okmDashboard = ContextWrapper.getContext().getBean(OKMDashboard.class
   Calendar cal = Calendar.getInstance();
   okmDashboard.visiteNode(null, "LastWeekTopDownloadedDocuments", "/okm:root/documen
  } catch (Exception e) {
   e.printStackTrace();
  }
 }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMDocument

Basics

*On almost methods you'll see parameter named "**docId**". The value of this parameter can be some valid document **UUID** node.*

Example of docId:

- *Using UUID -> "**c41f9ea0-0d6c-45da-bae4-d72b66f42d0f**";*

*About the parameter named "**fldId**", the value of this parameter can be some valid folder **UUID or** record node.*

Example of fldId:

- *Using UUID -> "**c6785440-0d6c-45da-1234-d9874563d0f**";*

*About the parameter named "**dstId**", the value of this parameter can be some valid folder **UUID or** record node.*

Example of dstId:

- *Using UUID -> "**c4455667-0d6c-45da-1234-d98564323e0**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

create

Description:

Method	*Return values*	*Description*
create(String token, Document doc, InputStream is)	***Document***	*Create a new document.*
The parameter doc must be initialized with the OpenKM path value into the repository. Optionally you can also set		

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

keywords and title values.

The other parameteres are not taken in consideration for document creation.

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            is = new FileInputStream("/home/openkm/sample.pdf");
            Document doc = new Document();
            doc.setPath("/okm:root/sample.pdf");
            // Optional
            doc.setTitle("sample document");
            doc.getKeywords().add("key1");
            doc.getKeywords().add("key2");
            // Create document
            doc = okmDocument.create(null, doc, is);
            System.out.println(doc);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

createSimple

Description:

Method	*Return values*	*Description*
createSimple(String token, String docPath, InputStream is)	***Document***	*Create a new document.*
The parameter docPath must be initialized with the OpenKM path value into the repository.		

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument

            is = new FileInputStream("/home/openkm/sample.pdf");
            String docPath = "/okm:root/sample.pdf";
            Document doc = okmDocument.createSimple(null, docPath, is);
            System.out.println(doc);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, String docId)	***void***	*Delete a document.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.delete(null, "d50133e3-dbfa-4d01-a109-28785cd48f40");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String docId)	***Document***	*Return a document properties.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            Document doc = okmDocument.getProperties(null, "3887ae75-dd67-4019-b4d1-2
            System.out.println(doc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getContent

Description:

Method	*Return values*	*Description*
getContent(String token, String docId, boolean checkout)	***InputStream***	*Return the document content.*
When paramer checkout is true, the document is marked for edition.		

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        InputStream is = null;
        OutputStream fos = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            is = okmDocument.getContent(null, "3887ae75-dd67-4019-b4d1-2f6962815403",
            File tmp = new File("/home/openkm/sample.pdf");
            fos = new FileOutputStream(tmp);
            IOUtils.copy(is, fos);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
            IOUtils.closeQuietly(fos);
        }
    }
}
```

getContentByVersion

Description:

Method	*Return values*	*Description*
getContentByVersion(String token, String docId, String versionId)	***InputStream***	*Return the document content by version.*
The parameter versionIs is the name of the document version.		

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        OutputStream fos = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            is = okmDocument.getContentByVersion(null, "f3a977f7-abfc-4f58-a033-98890
            File tmp = new File("/home/openkm/sample.pdf");
            fos = new FileOutputStream(tmp);
            IOUtils.copy(is, fos);
        } catch (Exception e) {
            e.printStackTrace();
```

```
        } finally {
            IOUtils.closeQuietly(is);
            IOUtils.closeQuietly(fos);
        }
    }
}
```

getExtractedText

Description:

Method	*Return values*	*Description*
getExtractedText(String token, String docId)	***String***	*Return the extracted text of the document.*

When document is upload to OpenKM goes into search engine queue to be processed. Until the process be finished this method will return an empty string.

From Database query view you can execute a query to verify is a document has yet been processed:

```
SELECT NDC_TEXT_EXTRACTED FROM OKM_NODE_DOCUMENT where NBS_UUID='1048eb46-2f32-4011-90e4-e084d838a592'
```

NBS_UUID value is the document UUID value. You can get it from properties tab in desktop view.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            String indexedText = okmDocument.getExtractedText(null, "f3a977f7-abfc-4f
            System.out.println(indexedText);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildren

Descripcion:

Method	*Return values*	*Description*
getChildren(String token, String		

fldId)	***List<Document>***	*Returns a list of all documents which their parent is fldId.*
The parameter fldId can be a folder or a record node.		

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            for (Document doc : okmDocument.getChildren(null, "b67fe3ae-bcca-4947-b8e
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

rename

Description:

Method	*Return values*	*Description*
rename(String token, String docId, String newName)	***Document***	*Changes the name of a document.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            Document doc = okmDocument.rename(null, "f3a977f7-abfc-4f58-a033-988903b3
            System.out.println(doc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

setProperties

Description:

Method	*Return values*	*Description*
setProperties(String token, Document doc)	***void***	*Changes some document properties.*

Variables allowed to be changed:

- *Title*
- *Description*
- *Language*
- *Associated categories*
- *Associated keywords*

Only not null and not empty variables will be taken in consideration.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            Document doc = okmDocument.getProperties(null, "f3a977f7-abfc-4f58-a033-9
            doc.getKeywords().add("key1");
            doc.setDescription("some description");
            okmDocument.setProperties(null, doc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setLanguage

Description:

Method	*Return values*	*Description*

setLanguage(String token, String docId, String lang)	***void***	*Sets the document language.*

The parameter lang must be ISO 691-1 compliant.

More information at: https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.setLanguage(null, "f3a977f7-abfc-4f58-a033-988903b31473", "en
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTitle

Description:

Method	*Return values*	*Description*
setTitle(String token, String docId, String title)	***void***	*Sets the document title.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.setTitle(null, "f3a977f7-abfc-4f58-a033-988903b31473", "Docum
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

checkout

Description:

Method	*Return values*	*Description*
checkout(String token, String docId)	***void***	*Marks the document for edition.*
Only one user can modify a document at a time. *Before starting edition you must do a checkout action that locks the edition process for other users and allows edition only to the user who has executed the action.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.checkout(null, "f3a977f7-abfc-4f58-a033-988903b31473");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

cancelCheckout

Description:

Method	*Return values*	*Description*
cancelCheckout(String token, String docId)	***void***	*Cancels a document edition.*
This action can only be done by the user who previously executed the checkout action.		

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.cancelCheckout(null, "f3a977f7-abfc-4f58-a033-988903b31473");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forceCancelCheckout

Description:

Method	*Return values*	*Description*
forceCancelCheckout(String token, String docId)	***void***	*Cancels a document edition.*

This method allows to cancel edition on any document.

It is not mandatory to execute this action by the same user who previously executed the checkout action.

This action can only be done by a super user (user with ROLE_ADMIN).

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.forceCancelCheckout(null, "f3a977f7-abfc-4f58-a033-988903b314
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isCheckedOut

Description:

Return

Method	values	Description
isCheckedOut(String token, String docId)	***void***	*Returns a boolean that indicates if the document is on edition or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            System.out.println(okmDocument.isCheckedOut(null, "f3a977f7-abfc-4f58-a03
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

checkin

Description:

Method	Return values	Description
checkin(String token, String docId, InputStream is, String comment)	***Version***	*Updates a document with a new version and returns an object with new Version values.*

Only the user who started the edition - checkout - is allowed to update the document.

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            is = new FileInputStream("/home/openkm/sample.pdf");
            okmDocument.checkin(null, "f3a977f7-abfc-4f58-a033-988903b31473", is, "op
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

checkin

Description:

Method	*Return values*	*Description*
checkin(String token, String docId, InputStream is, String comment, int increment)	***Version***	*Updates a document with a new version and returns an object with new Version values.*

The increment value depends on VersionNumeration implementation. Common values are:

- *VersionNumerationAdapter.INCREASE_DEFAULT for default.*
- *VersionNumerationAdapter.INCREASE_MINOR for increasing a minor version.*
- *VersionNumerationAdapter.INCREASE_MAJOR for increasing a major version.*
- *VersionNumerationAdapter.INCREASE_MAJOR_MINOR for increasing a major-minor version.*

More information about VersionNumeration at Creating your own Version Number Adapter.

Only the user who started the edition - checkout - is allowed to update the document.

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.plugin.vernum.VersionNumerationAdapter;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            is = new FileInputStream("/home/openkm/sample.pdf");
            okmDocument.checkin(null, "f3a977f7-abfc-4f58-a033-988903b31473", is, "op
                    VersionNumerationAdapter.INCREASE_MINOR);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

lock

Description:

Method	*Return values*	*Description*
lock(String token, String docId)	***LockInfo***	*Locks a document and returns an object with the Lock information.*

Only the user who locked the document is allowed to unlock.

A locked document cannot be modified by other users.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.LockInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            LockInfo lockInfo = okmDocument.lock(null, "f3a977f7-abfc-4f58-a033-98890
            System.out.println(lockInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unlock

Description:

Method	Return values	Description
unlock(String token, String docId)	***LockInfo***	*Unlocks a locked document.*

Only the user who locked the document is allowed to unlock.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.unlock(null, "f3a977f7-abfc-4f58-a033-988903b31473");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forceUnlock

Description:

Method	Return values	Description
forceUnlock(String token, String docId)	***LockInfo***	*Unlocks a locked document.*

This method allows to unlock any locked document.

It is not mandatory execute this action by the same user who previously executed the checkout lock action.

This action can only be done by a super user (user with ROLE_ADMIN).

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.forceUnlock(null, "f3a977f7-abfc-4f58-a033-988903b31473");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isLocked

Description:

Method	*Return values*	*Description*
isLocked(String token, String docId)	***void***	*Returns a boolean that indicates if the document is locked or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            System.out.println(okmDocument.isLocked(null, "f3a977f7-abfc-4f58-a033-98
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLockInfo

Description:

Method	*Return values*	*Description*
getLockInfo(String token, String docId)	***LockInfo***	*Returns an object with the Lock information.*

Example:

```
package com.openkm;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.api.OKMDocument;
import com.openkm.bean.LockInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            LockInfo lockInfo = okmDocument.getLockInfo(null, "f3a977f7-abfc-4f58-a03
            System.out.println(lockInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

purge

Description:

Method	*Return values*	*Description*
purge(String token, String docId)	***void***	*The document is definitely removed from repository.*

Usually you will purge documents into /okm:trash/userId - the personal trash user locations - but is possible to directly purge any document from the whole repository.

> *When a document is purged only will be able to be restored from a previously repository backup. The purge action removes the document definitely from the repository.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.purge(null, "f3a977f7-abfc-4f58-a033-988903b31473");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

move

Description:

Method	*Return values*	*Description*
move(String token, String docId, String dstId)	***void***	*Move document into some folder or record.*
The values of the dstId parameter should be a folder or record UUID.		

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.move(null, "f3a977f7-abfc-4f58-a033-988903b31473", "b67fe3ae-
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String docId, String dstId)	***void***	*Copy a document into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

Only the binary data and the security grants are copied to destination, the metadata, keywords, etc. of the document are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.copy(null, "f3a977f7-abfc-4f58-a033-988903b31473", "b67fe3ae-
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String docId, String dstId, String newName)	***void***	*Copy a document into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

When the parameter newName value is null,the document will preservate the same name.

Only the binary data and the security grants are copied to destination, the metadata, keywords, etc. of the document are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.copy(null, "f3a977f7-abfc-4f58-a033-988903b31473", "b67fe3ae-
                    "samples_renamed.pdf");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

extendedCopy

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String docId, String dstId, String docName, ExtendedAttributes extAttr)	***void***	*Copies a document with associated data into some folder or record.*

The values of the dstId parameter should be a folder or a record UUID.

When the parameter docName value is null, the document will preserve the same name.

By default only the binary data and the security grants, the metadata, keywords, etc. of the document are not copied.

Additional:

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original value of the notes will be copied.*
- *When wiki parameter is true the original value of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            ExtendedAttributes extAttr = new ExtendedAttributes();
            // copy keywords
            extAttr.setKeywords(true);
            extAttr.setCategories(true);
            extAttr.setNotes(true);
            extAttr.setPropertyGroups(true);

            okmDocument.extendedCopy(null, "f3a977f7-abfc-4f58-a033-988903b31473", "b
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getVersionHistorySize

Description:

Method	*Return values*	*Description*
getVersionHistorySize(String token, String docId)	***long***	*Returns the sum in bytes of all documents into documents history.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FormatUtil;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            System.out.println(FormatUtil.formatSize(okmDocument.getVersionHistorySiz
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isValid

Description:

Method	*Return values*	*Description*
isValid(String token, String docId)	***boolean***	*Returns a boolean that indicates if the node is a document or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
```

```
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            boolean valid = okmDocument.isValid(null, "f3a977f7-abfc-4f58-a033-988903
            System.out.println(valid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPath

Description:

Method	*Return values*	*Description*
getPath(String token, String uuid)	***String***	*Converts a document UUID to document path.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            String docPath = okmDocument.getPath(null, "f3a977f7-abfc-4f58-a033-98890
            System.out.println(docPath);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getDetectedLanguages

Description:

Method	*Return values*	*Description*
getDetectedLanguages(String token)	***List<String>***	*Return a list of available document languages what OpenKM can identify.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            for (String languages : okmDocument.getDetectedLanguages(null)) {
                System.out.println(languages);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createFromTemplate

Description:

Method	*Return values*	*Description*
createFromTemplate(String token, String docId, String dstPath, Map<String, String> properties, ExtendedAttributes attributes)	***Document***	*Creates a new document from template and returns an object Document.*

The dstPath can be a folder or a record UUID.

The parameter language is optional.

When the template uses metadata groups to fill fields into, then these values are mandatory and must be set into properties parameter.

For more information about Templates and metadata check: Creating templates.

Additional attribute parameters:

- *When category parameter is true the original values of the categories will be copied.*
- *When keywords parameter is true the original values of the keywords will be copied.*
- *When propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When notes parameter is true the original values of the notes will be copied.*
- *When wiki parameter is true the original values of the wiki will be copied.*

Example:

The example below is based on Creating PDF template sample.

```
package com.openkm;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.HashMap;
import java.util.Map;
import java.util.List;

import com.google.gson.Gson;
import com.openkm.api.OKMDocument;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;
import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            Map<String, String> properties = new HashMap<>();
            properties.put("okp:tpl.name", "Some name");
            Calendar cal = Calendar.getInstance();
            // Value must be converted to String ISO 8601 compliant
            properties.put("okp:tpl.bird_date", ISO8601.formatBasic(cal));

            List<String> list = new ArrayList<>();
            list.add("java");
            properties.put("okp:tpl.language", new Gson().toJson(list).toString());

            okmDocument.createFromTemplate(null, "86633fe8-1b0d-48ac-a716-7685022adad
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateFromTemplate

Description:

Method	*Return values*	*Description*
updateFromTemplate(String token, String docId, String dstId, Map<String, String> properties)	***void***	*Updates a document previously created from the template and returns an object Document.*

This method only has sense when the template uses metadata groups to fill fields in.

For more information about Templates and metadata check: Creating templates .

Example:

The example below is based on Creating PDF template sample.

```
package com.openkm;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.google.gson.Gson;
import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;
import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            Map<String, String> properties = new HashMap<>();
            properties.put("okp:tpl.name", "Some name");
            Calendar cal = Calendar.getInstance();
            // Value must be converted to String ISO 8601 compliant
            properties.put("okp:tpl.bird_date", ISO8601.formatBasic(cal));

            List<String> list = new ArrayList<>();
            list.add("java");
            properties.put("okp:tpl.language", new Gson().toJson(list).toString());

            okmDocument.updateFromTemplate(null, "86633fe8-1b0d-48ac-a716-7685022adad
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setAnnotations

Description:

Method	*Return values*	*Description*
setAnnotations(String token, String docId, String verName, String annotation)	***void***	*Set document annotations of some document version.*

Example:

```
package com.openkm;
import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.setAnnotations(null, "3887ae75-dd67-4019-b4d1-2f6962815403",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAnnotations

Description:

Method	*Return values*	*Description*
getAnnotations(String token, String docId, String verName)	***String***	*Return the document annotations of some document version.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            String annotation = okmDocument.getAnnotations(null, "3887ae75-dd67-4019-
            System.out.println(annotation);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getCheckedOut

Description:

Method	*Return values*	*Description*
getCheckedOut(String token)	***List<Document>***	*Return the list of all documents checkout in the repository*

Example:

```
package com.openkm;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.api.OKMDocument;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            for (Document doc : okmDocument.getCheckedOut(null)) {
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

stamp

Description:

Method	*Return values*	*Description*
stamp(String token, String docId, int type, long stId)	***void***	*Stamp a document.*

The parameter type indicates stamp type:

- *"text" with value 0;*
- *"image" with value 1;*

The parameter stId is the unique identifier value of the stamp definition. You can get it from Administration > Stamp view.

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            int type = 0; // indicate text stamp definition
            long stId = 1; // unique identifier of the stamp definition
            okmDocument.stamp(null, "3887ae75-dd67-4019-b4d1-2f6962815403", type, stI
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

setDescription

Description:

Method	*Return values*	*Description*
setDescription(String token, String docId, String description)	***void***	*Set a description.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            okmDocument.setDescription(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setNodeClass

Description:

Method	*Return values*	*Description*
setNodeClass(String token, String docId, long ncId)	***void***	*Set the NodeClass.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            long ncId = 2;
            okmDocument.setNodeClass(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b", nc
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

setDispositionStage

Description:

Method	*Return values*	*Description*
setDispositionStage(String token, String docId, long stage)	***void***	*Set the disposition stage*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            long stage = 1;
            okmDocument.setDispositionStage(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd36
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createWizard

Description:

Method	*Return values*	*Description*
createWizard(String token, String docPath, long nodeClass, InputStream is)	***WizardNode***	*Create a new document with wizard.*

The parameters ***docPath*** *should be any valid folder or record* ***PATH***.

> *The WizardNode contains a list of pending actions what should be done to complete the process of document creation. These might be:*
>
> - *Add keyword*
> - *Add Categories*
> - *Add Metadata*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.bean.WizardNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            InputStream is = new FileInputStream("/home/gnujavasergio/okm/logo.png");
            WizardNode wn = okmDocument.createWizard(null, "/okm:root/test/logo.png",
            System.out.print(wn);
            IOUtils.closeQuietly(is);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNumberOfPages

Description:

Method	*Return values*	*Description*
getNumberOfPages(String token, String uuid)	***int***	*Get the number of pages of a document.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            System.out.println("The number of pages is : " + okmDocument.getNumberOfP
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPageAsImage

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
getPageAsImage(String token, String uuid, int pageNumber, int maxWidth, int maxHeight)	***String***	*Return specific page as an image encoded as a String in B64.*

Example:

```
package com.openkm;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            System.out.println(okmDocument.getPageAsImage(null, "3fe350e2-69e8-4681-a
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

mergePdf

Description:

Method	*Return values*	*Description*
mergePdf(String token, String destinationPath, String docName, List<String> paths)	***void***	*Merge two or more PDF files.*
*The parameters **destinationPath** should be any valid folder or record PATH.*		

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;

public class Test {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    public static void main(String[] args) {
        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            List<String> uuids = new ArrayList<>();
            uuids.add("/okm:root/test/document.pdf");
            uuids.add("/okm:root/test/document1.pdf");
            String fldUuid = "/okm:root/mergue";
            okmDocument.mergePdf(null, fldUuid, "MergePdf.pdf", uuids);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

OKMFolder

Basics

*On almost methods you'll see parameter named "**fldId**". The value of this parameter can be some valid folder **UUID**.*

Example of fldId:

- *Using UUID -> "**b67fe3ae-bcca-4947-b8ea-0cc7aedd4474**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

create

Description:

Method	*Return values*	*Description*
create(String token, Folder fld)	***Folder***	*Creates a new folder and returns as a result an object Folder.*

*The variable **path** into the parameter **fld**, must be initializated. It indicates the folder path into OpenKM.*

```
Folder fld = new Folder();
fld.setPath("/okm:root/test");
```

The other variables of the Folder (fld) will not take any effect on the folder creation.

We suggest using the method below to create the Folder Simply rather this one.

Example:

```
package com.openkm;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.api.OKMFolder;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            Folder fld = new Folder();
            fld.setPath("/okm:root/test");
            okmFolder.create(null, fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createSimple

Description:

Method	*Return values*	*Description*
createSimple(String token, String fldPath)	***Folder***	*Creates a new folder and returns as a result an object Folder.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            String fldPath = "/okm:root/test";
            okmFolder.createSimple(null, fldPath);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String fldId)	***Folder***	*Return the folder properties.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            Folder fld = okmFolder.getProperties(null, "9f64248c-e049-4706-b89e-6d725
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, String fldId)	***void***	*Delete a folder.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.delete(null, "9f64248c-e049-4706-b89e-6d725842c7f7");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

purge

Description:

Method	*Return values*	*Description*

purge(String token, String fldId)	***void***	*The folder is definitely removed from the repository.*

Usually you will purge folders into /okm:trash/userId - the personal trash user locations - but it is possible to directly purge any folder from the whole repository.

When a folder is purged, it will only be able to be restored from a previously repository backup. The purge action removes the folder definitely from the repository.

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.delete(null, "9f64248c-e049-4706-b89e-6d725842c7f7");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

rename

Description:

Method	*Return values*	*Description*
rename(String token, String fldId, String newName)	***Folder***	*Rename a folder.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.rename(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "newname")
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

move

Description:

Method	*Return values*	*Description*
move(String token, String fldId, String dstId)	***void***	*Moves folder into some folder or record*
The values of the dstId parameter should be a folder or record UUID.		

Example:

```
package com.openkm;
import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.move(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "b67fe3ae-bc
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String fldId, String dstId)	***void***	*Copies a folder into a folder or record.*
The values of the dstId parameter should be a folder or record UUID.		

Only the security grants are copied to destination, the metadata, keywords, etc. of the folder are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.copy(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "b67fe3ae-bc
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String fldId, String dstId, String newName)	***void***	*Copies a folder into a folder or record.*

The values of the dstId parameter should be a folder or record UUID.

When parameter newName value is null, folder will preservate the same name.

Only the security grants are copied to destination, the metadata, keywords, etc. of the folder are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.copy(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "b67fe3ae-bc
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

extendedCopy

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String fldId, String dstId, ExtendedAttributes extAttr)	***void***	*Copies a folder with the associated data into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

By default only the binary data and the security grants, the metadata, keywords, etc. of the folder are not copyed.

Additional extended attributes parameters:

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original values of the notes will be copied.*
- *When the wiki parameter is true the original values of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class

            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setKeywords(true);
            extAttr.setNotes(true);
            extAttr.setCategories(true);
            extAttr.setPropertyGroups(true);

            okmFolder.extendedCopy(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "b67
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

extendedCopy

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String fldId, String dstId, ExtendedAttributes extAttr, String newName)	***void***	*Copies a folder with the associated data into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

When parameter newName value is null, folder will preservate the same name.

By default only the binary data and the security grants, the metadata, keywords, etc. of the folder are not copyed.

Additional extended attributes parameters:

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original values of the notes will be copied.*
- *When the wiki parameter is true the original values of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class

            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setKeywords(true);
            extAttr.setNotes(true);
            extAttr.setCategories(true);
            extAttr.setPropertyGroups(true);
```

```
            okmFolder.extendedCopy(null, "7aceb86c-1dc8-4551-aebd-b7bd73d01263", "b67
                    extAttr, "newname");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildren

Description:

Method	*Return values*	*Description*
getChildren(String token, String fldId)	***List<Folder>***	*Returns a list of all folders which their parent is fldId.*
The parameter fldId can be a folder or a record node.		

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            for (Folder fld : okmFolder.getChildren(null, "b67fe3ae-bcca-4947-b8ea-0c
                System.out.println(fld);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getContentInfo

Description:

Method	*Return values*	*Description*
getContentInfo(String token, String fldId)	***ContentInfo***	*Return and object ContentInfo with information about folder.*
The ContentInfo object retrives information about:		

- *The number of folders into.*
- *The number of documents into.*
- *The number of records into.*
- *The number of mails into.*
- *The size in bytes of all objects into the folder.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.bean.ContentInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            ContentInfo ci = okmFolder.getContentInfo(null, "b67fe3ae-bcca-4947-b8ea-
            System.out.println(ci);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isValid

Description:

Method	*Return values*	*Description*
isValid(String token, String fldId)	***boolean***	*Returns a boolean that indicates if the node is a folder or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            boolean valid = okmFolder.isValid(null, "7aceb86c-1dc8-4551-aebd-b7bd73d0
            System.out.println(valid);
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

getPath

Description:

Method	*Return values*	*Description*
getPath(String token, String uuid)	***String***	*Convert folder UUID to folder path.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            String path = okmFolder.getPath(null, "7aceb86c-1dc8-4551-aebd-b7bd73d012
            System.out.println(path);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setStyle

Description:

Method	*Return values*	*Description*
setStyle(String token, String fldId, long styleId)	***void***	*Set the folder style.*

More information at Folder style.

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.setStyle(null, "/okm:root/test", 1);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createMissingFolders

Description:

Method	*Return values*	*Description*
createMissingFolders(String token, String fldPath)	***void***	*Create missing folders based on all the path nodes.*

> *The method checks all subfolders and when detects some one is missing, it creates all missing subfolders from there.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.createMissingFolders(null, "/okm:root/test/missing/from/here");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addNodeClassChildren

Description:

Method	*Return values*	*Description*

addNodeClassChildren(String token, String fldId, long ncId)	***void***	*Add the NodeClass.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            long ncId = 2;
            okmFolder.addNodeClassChildren(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setDescription

Description:

Method	*Return values*	*Description*
setDescription(String token, String uuid, String description)	***void***	*Set a description.*

Example:

```
package com.openkm;

import com.openkm.api.OKMFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMFolder okmFolder = ContextWrapper.getContext().getBean(OKMFolder.class
            okmFolder.setDescription(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b", "s
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMImport

Basics

*On almost methods you'll see parameter named "**parentUuid**". The value of this parameter can be some valid document **UUID** node.*

Example of docId:

- *Using UUID -> "**c41f9ea0-0d6c-45da-bae4-d72b66f42d0f***

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

createDocument

Description:

Method	*Return values*	*Description*
createDocument(String token, String parentUuid, String docName, File file)	***NodeDocument***	*Create a new document.*
*The parameter **parentUuid** must be a **valid folder or record UUID**.*		

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMImport;
import com.openkm.db.bean.NodeDocument;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMImport okmImport = ContextWrapper.getContext().getBean(OKMImport.class
            File doc = new File("/home/openkm/sample.pdf");
            NodeDocument nodeDoc = okmImport.createDocument(null, "0a19e616-1d11-4ef7
            System.out.println(nodeDoc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createFolder

Description:

Method	*Return values*	*Description*
createDocument(String token, String parentUuid, String fldName)	***NodeFolder***	*Create a new folder.*
*The parameter **parentUuid** must be a **valid folder or record UUID**.*		

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMImport;
import com.openkm.db.bean.NodeFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMImport okmImport = ContextWrapper.getContext().getBean(OKMImport.class
            NodeFolder nodeFld = okmImport.createFolder(null, "0a19e616-1d11-4ef7-a7c
            System.out.println(nodeFld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

importFolder

Description:

		Return

Method	*values*	*Description*
importFolder(String token, String parentUuid, File folder, int threads)	***void***	*Import local folder hierarchy.*

*The parameter **parentUuid** must be a **valid folder or record UUID**.*

*The value **threads** set the **number of concurrent threads** used by import feature. This value should **never be upper than the number of the server cores**. If you have 20 cores a good starting point might be use 10 threads, take in consideration other software like database, OS etc... are also using the hardware.*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMImport;
import com.openkm.db.bean.NodeFolder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMImport okmImport = ContextWrapper.getContext().getBean(OKMImport.class
            File fld = new File("/home/openkm/import/");
            okmImport.importFolder(null, "0a19e616-1d11-4ef7-a7c7-f1e109f8ea3c", fld,
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMLink

Methods

create

Description:

Method	*Return values*	*Description*
create(String token, String lnkPath, String nodeId)	***Node***	*Create a link for a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMLink;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMLink okmLink = ContextWrapper.getContext().getBean(OKMLink.class);
            Node node = okmLink.create(null, "/okm:root/link", "3e4ce783-bdf7-41f1-ae
            System.out.println(node);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildren

Description:

Method	*Return values*	*Description*
getChildren(String token, String fldId, String type)	***List<Node>***	*Returns a list of all the links by type which their parent is fldId.*

The values of the fldId parameter should be a folder or a record UUID.

The paramenter type have the next Available values:

- *Document.Type*
- *Folder.Type*

- *Mail.Type*
- *Record.Type*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMLink;
import com.openkm.bean.Document;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMLink okmLink = ContextWrapper.getContext().getBean(OKMLink.class);
            List<Node> nodes = okmLink.getChildren(null, "9f2d7c6c-18e4-4137-9a21-dd3
            for (Node node : nodes) {
                System.out.println(node);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLinks

Description:

Method	*Return values*	*Description*
getLinks(String token, nodeId)	***List<Node>***	*Returns a list of all the links of the node.*

Example

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMLink;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMLink okmLink = ContextWrapper.getContext().getBean(OKMLink.class);
            List<Node> nodes = okmLink.getLinks(null, "3e4ce783-bdf7-41f1-ae61-6aab23
            for (Node node : nodes) {
                System.out.println(node);
```

```
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isValid

Description:

Method	*Return values*	*Description*
isValid(String token, String lnkId)	***boolean***	*Returns a boolean that indicates if the node is a link or not.*

Example

```
package com.openkm;

import com.openkm.api.OKMLink;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMLink okmLink = ContextWrapper.getContext().getBean(OKMLink.class);
            boolean valid = okmLink.isValid(null, "1515adfb-ab2f-4209-b1ab-57fa9f4bb6
            System.out.println(valid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMMail

Basics

*On almost methods you'll see parameter named "**mailId**". The value of this parameter can be some valid mail **UUID**.*

Example of mailId:

- *Using UUID -> "**0747c0cb-5e4a-4088-92a9-ab97b401af75**";*

*About the parameter named "**fldId**", the value of this parameter can be some valid folder **UUID** or record node.*

Example of fldId:

- *Using UUID -> "**b67fe3ae-bcca-4947-b8ea-0cc7aedd4474**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String mailId)	***Mail***	*Returns the mail properties.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            Mail mail = okmMail.getProperties(null, "0747c0cb-5e4a-4088-92a9-ab97b401
            System.out.println(mail);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createAttachment

Description:

Method	*Return values*	*Description*
createAttachment(String token, String mailId, String docName, InputStream is)	***Document***	*Add a attachment into the mail.*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.poi.util.IOUtils;

import com.openkm.api.OKMMail;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            is = new FileInputStream("/home/openkm/sample.pdf");
            Document doc = okmMail.createAttachment(null, "0747c0cb-5e4a-4088-92a9-ab
            System.out.println(doc);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

deleteAttachment

Description:

Method	*Return values*	*Description*

deleteAttachment(String token, String mailId, String docId)	***void***	*Delete a attachment into the mail.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.deleteAttachment(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "c
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAttachments

Description:

Method	*Return values*	*Description*
getAttachments(String token, String mailId)	***List<Document>***	*Returns a list of all attachments of a mail.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            for (Document doc : okmMail.getAttachments(null, "0747c0cb-5e4a-4088-92a9
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

delete

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
delete(String token, String mailId)	**void**	Delete a mail.

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.delete(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

purge

Description:

Method	Return values	Description
purge(String token, String mailId)	**void**	The mail is definitely removed from the repository.

Usually you will purge mails into /okm:trash/userId - the personal trash user locations - but it is possible to directly purge any mail from the whole repository.

When a mail is purged, it will only be able to be restored from a previously repository backup. The purge action removes the mail definitely from the repository.

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.purge(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75");
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

rename

Description:

Method	*Return values*	*Description*
rename(String token, String mailId, String newName)	***Mail***	*Rename a mail.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.rename(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "newname");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

move

Description:

Method	*Return values*	*Description*
move(String token, String mailId, String dstId)	***void***	*Moves mail into some folder or record*
The values of the dstId parameter should be a folder or record UUID.		

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.move(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "3636f3bb-779c
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String mailId, String dstId)	***void***	*Copies a mail into a folder or record.*

The values of the dstId parameter should be a folder or record UUID.

Only the security grants are copied to destination, the metadata, keywords, etc. of the mail are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.copy(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "3636f3bb-779c
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String mailId, String dstId, String	***void***	*Copies a mail into a folder or*

newName)		*record.*

The values of the dstId parameter should be a folder or record UUID.

When parameter newName value is null, mail will preservate the same name.

> *Only the security grants are copied to destination, the metadata, keywords, etc. of the mail are not copied.*
>
> *See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.copy(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "3636f3bb-779c
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

extendedCopy

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String mailId, String dstId, ExtendedAttributes extAttr)	***void***	*Copies a mail with the associated data into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

> *By default only the binary data and the security grants, the metadata, keywords, etc. of the mail are not copyed.*
>
> *Additional extended attributes parameters:*
>
> - *When the category parameter is true the original values of the categories will be copied.*
> - *When the keywords parameter is true the original values of the keywords will be copied.*

- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original values of the notes will be copied.*
- *When the wiki parameter is true the original values of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setNotes(true);
            extAttr.setKeywords(true);
            extAttr.setCategories(true);
            extAttr.setPropertyGroups(true);
            okmMail.extendedCopy(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "3636f
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

extendedCopy

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String mailId, String dstId, ExtendedAttributes extAttr, String newName)	***void***	*Copies a mail with the associated data into some folder or record.*

The values of the dstId parameter should be a folder or record UUID.

When parameter newName value is null, mail will preservate the same name.

By default only the binary data and the security grants, the metadata, keywords, etc. of the mail are not copyed.

Additional extended attributes parameters:

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original values of the notes will be copied.*
- *When the wiki parameter is true the original values of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setNotes(true);
            extAttr.setKeywords(true);
            extAttr.setCategories(true);
            extAttr.setPropertyGroups(true);
            okmMail.extendedCopy(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "3636f
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildren

Description:

Method	*Return values*	*Description*
getChildren(String token, String fldId)	***List<Mail>***	*Returns a list of all mails which their parent is mailId.*
The parameter mailId can be a folder or a record node.		

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            for (Mail mail : okmMail.getChildren(null, "3636f3bb-779c-4851-b145-0d53e
                System.out.println(mail);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isValid

Description:

Method	*Return values*	*Description*
isValid(String token, String mailId)	***boolean***	*Returns a boolean that indicates if the node is a mail or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            boolean valid = okmMail.isValid(null, "0747c0cb-5e4a-4088-92a9-ab97b401af
            System.out.println(valid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPath

Description:

Method	*Return values*	*Description*
getPath(String token, String uuid)	***String***	*Convert folder UUID to mail path.*

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            String path = okmMail.getPath(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75
            System.out.println(path);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendMail

Description:

Method	*Return values*	*Description*
sendMail(String token, List<String> recipients, String subject, String body)	***void***	*Sends a mail.*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            StringBuffer body = new StringBuffer();
            body.append("Test message body.");
            List<String> recipients = new ArrayList<>();
            recipients.add("gnu.java.sergio@gmail.com");
            okmMail.sendMail(null, recipients, "Testing sending mail from OpenKM", bo
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTitle

Description:

Method	*Return values*	*Description*

setTitle(String token, String mailId, String title)	***void***	*Sets the mail title.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.setTitle(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "new title
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendMailWithAttachments

Description:

Method	*Return values*	*Description*
sendMailWithAttachments(String token, List<String> toRecipients, List<String> ccRecipients, List<String> bccRecipients, List<String> replyToMails, String subject, String body, List<String> docsId, String dstId)	***Mail***	*Send mail message with attachement.*

The values of the dstId parameter should be a folder or record ***UUID****. The dstId parameter indicate where the mail will be stored in the repository after is sent.*

Other parameters:

- *toRecipients are a list of mail accounts destination.*
- *subject is the mail subject.*
- *ccRecipients, bccRecipients are a list of mail accounts destination (optional)*
- *replyToMails are a list of mails accounts for replying (optional)*
- *docsId are a list of valid document UUID already into OpenKM that will be send as attachment into mail (optional).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            List<String> toRecipients = Arrays.asList("gnujavasergio@mail.com");
            List<String> ccRecipients = new ArrayList<>();
            List<String> bccRecipients = new ArrayList<>();
            List<String> replyToMails = new ArrayList<>();
            List<String> docsId = Arrays.asList("b153c4b7-3d1c-4589-bd42-0ed0f34fd338

            Mail mail = okmMail.sendMailWithAttachments(null, toRecipients, ccRecipie
                    "some subject", "some body", docsId, "b67fe3ae-bcca-4947-b8ea-0cc
            System.out.println(mail);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

importEml

Description:

Method	*Return values*	*Description*
importEml(String fldId, String title, InputStream is)	***Mail***	*Import a mail in EML format.*
The values of the fldId parameter should be a folder or record ***UUID****. The path parameter indicate where the mail will be stored in the repository after is imported.*		

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

 public static void main(String[] args) {
  InputStream is = null;
        try {
         OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            is = new FileInputStream("/home/openkm/sample1.eml");
            Mail mail = okmMail.importEml(null, "b67fe3ae-bcca-4947-b8ea-0cc7aedd4474
            System.out.println(mail);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
 }
}
```

importMsg

Description:

Method	*Return values*	*Description*
importMsg(String fldId, String title, InputStream is)	***Mail***	*Import a mail in MSG format.*

The values of the fldId parameter should be a folder or record ***UUID****. The path parameter indicate where the mail will be stored in the repository after is imported.*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            is = new FileInputStream("/home/openkm/test subject.msg");
            Mail mail = okmMail.importMsg(null, "b67fe3ae-bcca-4947-b8ea-0cc7aedd4474
            System.out.println(mail);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
```

```
}
```

setDocumentNodeClass

Description:

Method	*Return values*	*Description*
setNodeClass(String token, String uuid, long ncId)	***void***	*Set the NodeClass.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            long ncId = 2;
            okmMail.setNodeClass(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b", ncId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setDispositionStage

Description:

Method	*Return values*	*Description*
setDispositionStage(String token, String uuid, long stage)	***void***	*Set the disposition stage*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            long stage = 1;
            okmMail.setDispositionStage(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75",
        } catch (Exception e) {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
            e.printStackTrace();
        }
    }
}
```

setDescription

Description:

Method	*Return values*	*Description*
setDescription(String token, String uuid, String description)	***void***	*Set the description.*

Example:

```
package com.openkm;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            okmMail.setDescription(null, "0747c0cb-5e4a-4088-92a9-ab97b401af75", "som
            Mail mail = okmMail.getProperties(null, "b0747c0cb-5e4a-4088-92a9-ab97b40
            System.out.println(mail);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getContent

Description:

Method	*Return values*	*Description*
geContent(String token, String mailId)	***InputStream***	*Retrieve mail content (binary data).*

Example:

```
package com.openkm;

import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

import org.apache.commons.io.IOUtils;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            OutputStream fos = new FileOutputStream("/home/openkm/test.eml");
            InputStream is = okmMail.getContent(null, "b405a504-d8cb-4166-ac51-22f68a
            IOUtils.copy(is, fos);
            IOUtils.closeQuietly(is);
            IOUtils.closeQuietly(fos);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createWizard

Description:

Method	*Return values*	*Description*
createWizard(String token, String path, String title, InputStream is, String type)	***WizardNode***	*Create a new email with a wizard.*

*The parameters **path** should be any valid folder or record **PATH**.*

Available types:

- *Mail.ORIGIN_MSG*
- *Mail.ORIGIN_EML*

The WizardNode contains a list of pending actions what should be done to complete the process of document creation. These might be:

- *Add keyword*
- *Add Categories*
- *Add Metadata*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMMail;
import com.openkm.bean.Mail;
import com.openkm.bean.WizardNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            InputStream is = new FileInputStream("/home/openkm/sample2.eml");
            WizardNode wn = okmMail.createWizard(null, "/okm:root/test", "test title"
            System.out.print(wn);
            IOUtils.closeQuietly(is);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forwardEmail

Description:

Method	*Return values*	*Description*
forwardEmail(String token, String uuid, List<String> users, List<String> roles, List<String> mails, String message)	***void***	*Forward an email to a list of users, roles or external emails.*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMMail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMMail okmMail = ContextWrapper.getContext().getBean(OKMMail.class);
            String mailUuid = "f123a950-0329-4d62-8328-0ff500fd42db";
            List<String> users = new ArrayList<>();
            users.add("userTest");
            List<String> roles = new ArrayList<>();
            roles.add("ROLE_USER");
            List<String> mails = new ArrayList<>();
            mails.add("test@none.com");

            okmMail.forwardEmail(null, mailUuid, users, roles, mails, "Any message");
        } catch (Exception e) {
            e.printStackTrace();
```

```
                }
            }
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMNode

Basics

*On almost methods you'll see parameter named "**nodeId**". The value of this parameter can be some valid **UUID** node.*

Example of docId:

- *Using UUID -> "**c41f9ea0-0d6c-45da-bae4-d72b66f42d0f**";*

*About the parameter named "**fldId**", the value of this parameter can be some valid folder **UUID or** record node.*

Example of fldId:

- *Using UUID -> "**c6785440-0d6c-45da-1234-d9874563d0f**";*

*About the parameter named "**dstId**", the value of this parameter can be some valid folder **UUID or** record node.*

Example of dstId:

- *Using UUID -> "**c4455667-0d6c-45da-1234-d98564323e0**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

getNodeByUuid

Description:

Method	*Return values*	*Description*
getNodeByUuid(String token, String uuid)	***Node***	*Get a node by uuid.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            Node node = okmNode.getNodeByUuid(null, "78f9c377-e557-48c7-804e-35c0ca28
            System.out.println(node);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getVersionHistory

Description:

Method	*Return values*	*Description*
getVersionHistory(String token, String nodeId)	***List<Version>***	*Returns the version history list of a node.*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.Version;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            List<Version> versions = okmNode.getVersionHistory(null, "78f9c377-e557-4
            for (Version version : versions) {
                System.out.println(version);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

restoreVersion

Description:

Method	*Return values*	*Description*

restoreVersion(String token, String nodeId, String versionId)	***void***	*Restore the version of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.restoreVersion(null, "78f9c377-e557-48c7-804e-35c0ca284739", "1.2
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

purgeVersionHistory

Description:

Method	*Return values*	*Description*
purgeVersionHistory(String token, String nodeId)	***void***	*Purge the version history of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.purgeVersionHistory(null, "3767deb4-21e7-4272-82be-fece5384fbab")
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

mayBePromotedAsRecord

Description:

Method	*Return values*	*Description*

mayBePromotedAsRecord(String token, String nodeId, boolean fullEvaluation)	***PromoteAsRecordEvaluation***	*Returns the result of the evaluation to promote a node to the "record" status. When the node can not be promoted to the "record" status, the result of the evaluation contains information about the reason why can not be promoted.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.plugin.fileplan.record.bean.PromoteAsRecordEvaluation;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            PromoteAsRecordEvaluation pre = okmNode.mayBePromotedAsRecord(null, "3767
            System.out.println(pre);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

promoteAsRecord

Description:

Method	*Return values*	*Description*
promoteAsRecord(String token, String nodeId)	***void***	*Promote the node to the "record" status.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.promoteAsRecord(null, "3767deb4-21e7-4272-82be-fece5384fbab");
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

degradeRecord

Description:

Method	*Return values*	*Description*
degradeRecord(String token, String nodeId)	***void***	*Degrade a node with "record" status.*

This action only can be done by administrators.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.degradeRecord(null, "3767deb4-21e7-4272-82be-fece5384fbab");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isElectronicRecordPath

Description:

Method	*Return values*	*Description*
isElectronicRecordPath(String token, String nodeId)	***boolean***	*Returns true when the node is into an electronic record.*

Return true when one of the parents of the node is an electronic record.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            System.out.println(okmNode.isElectronicRecordPath(null, "3767deb4-21e7-42
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getElectronicRecordInPath

Description:

Method	*Return values*	*Description*
getElectronicRecordInPath(String token, String nodeId)	***Record***	*Get the electronic record in the path.*

Returns the first electronic record in the path of the node.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            Record record = okmNode.getElectronicRecordInPath(null, "3767deb4-21e7-42
            System.out.println(record);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildrenPaginated

Description:

		Return

Method	*values*	*Description*
getChildrenPaginated(String token, String nodeId, int offset, int limit, String filter, String orderByField, boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList)	***PageInfo***	*Get children nodes paginated.*

*The parameter "**limit**" and "**offset**" allows you to retrieve just a portion of the results of a query.*

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

*The parameter "**orderByFilter**" can have the following values name or created*

For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.PageInfo;
import com.openkm.db.bean.NodeBase;
import com.openkm.db.bean.NodeDocument;
import com.openkm.db.bean.NodeFolder;
import com.openkm.db.bean.NodeMail;
import com.openkm.db.bean.NodeRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            List<Class<? extends NodeBase>> filteredByNodeTypeList = new ArrayList<>(
            filteredByNodeTypeList.add(NodeDocument.class);
```

```
            filteredByNodeTypeList.add(NodeFolder.class);
            filteredByNodeTypeList.add(NodeMail.class);
            filteredByNodeTypeList.add(NodeRecord.class);

            // nodeId Folder and Record UUID or Path
            String nodeId = "39479efe-de5e-468e-91a7-24d2aa3f8837";
            String orderByField = "created";
            boolean orderAsc = true;
            PageInfo pageInfo = okmNode.getChildrenPaginated(null, nodeId, 0, 10, "",
                    filteredByNodeTypeList);
            System.out.println("Filtered Elements: " + pageInfo.getFilteredElements()
            System.out.println("Total Elements: " + pageInfo.getTotalElements());
            for (Object object : pageInfo.getObjects()) {
                NodeBase nodeBase = (NodeBase) object;
                System.out.println(nodeBase.getPath());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildrenByCategoryPaginated

Description:

Method	*Return values*	*Description*
getChildrenByCategoryPaginated(String token, String nodeId, int offset, int limit, String filter, String orderByField, boolean orderAsc, List<Class<? extends NodeBase>> filteredByNodeTypeList)	***PageInfo***	*Get children nodes by category paginated.*

*The parameter "**limit**" and "**offset**" allows you to retrieve just a portion of the results of a query.*

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

*The parameter "**orderByFilter**" can have the following values name or created*

For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*

- *offset=10*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.PageInfo;
import com.openkm.db.bean.NodeBase;
import com.openkm.db.bean.NodeDocument;
import com.openkm.db.bean.NodeFolder;
import com.openkm.db.bean.NodeMail;
import com.openkm.db.bean.NodeRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            List<Class<? extends NodeBase>> filteredByNodeTypeList = new ArrayList<>(
            filteredByNodeTypeList.add(NodeDocument.class);
            filteredByNodeTypeList.add(NodeFolder.class);
            filteredByNodeTypeList.add(NodeMail.class);
            filteredByNodeTypeList.add(NodeRecord.class);

            // nodeId Folder and Record UUID or Path
            String nodeId = "39479efe-de5e-468e-91a7-24d2aa3f8837";
            String orderByField = "created";
            boolean orderAsc = true;
            PageInfo pageInfo = okmNode.getChildrenByCategoryPaginated(null, nodeId,
                    filteredByNodeTypeList);
            System.out.println("Filtered Elements: " + pageInfo.getFilteredElements()
            System.out.println("Total Elements: " + pageInfo.getTotalElements());
            for (Object object : pageInfo.getObjects()) {
                NodeBase nodeBase = (NodeBase) object;
                System.out.println(nodeBase.getPath());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getBreadcrumb

Description:

Method	*Return values*	*Description*
getBreadcrumb(String token, String fldId)	***List<BreadCrumbItem>***	*Get breadcrumb.*

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.BreadCrumbItem;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            List<BreadCrumbItem> list = okmNode.getBreadcrumb(null, "39479efe-de5e-46
            for (BreadCrumbItem item : list) {
                System.out.println(item.getPath());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

subscribe

Description:

Method	*Return values*	*Description*
subscribe(String token, String nodeId)	***void***	*Adds a subscription to a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.subscribe(null, "39479efe-de5e-468e-91a7-24d2aa3f8837");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unsubscribe

Description:

Method	*Return values*	*Description*

unsubscribe(String token, String nodeId)	***void***	*Delete a subscription to a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.unsubscribe(null, "39479efe-de5e-468e-91a7-24d2aa3f8837");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

importZip

Description:

Method	*Return values*	*Description*
importZip(String token, String fldUuid, InputStream inputStream)	***void***	*Import a zip file.*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            InputStream is = new FileInputStream("/home/gnujavasergio/okm/import.zip"
            okmNode.importZip(null, "212e7c1f-443d-4aac-a12c-0b818ca03419", is);
            IOUtils.closeQuietly(is);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unzip

Description:

Method	Return values	Description
unzip(String token, String uuid, String destinationPath)	***void***	*Unzip file.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            String uuid = "82555dd1-bcc2-4e64-81cb-5f7c2b0d7801"; // zip File
            String destinationPath = "/okm:root/test"; // destination path
            okmNode.unzip(null, uuid, destinationPath);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

exportZip

Description:

Method	Return values	Description
exportZip(List<String> paths, boolean background)	***ByteArrayInputStream***	*Export as a zip file.*

Example:

```
package com.openkm;

import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.List;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
```

```
            List<String> uuids = new ArrayList<>();
            uuids.add("/okm:root/test/logo.png");
            uuids.add("/okm:root/test/invoice.pdf");

            OutputStream fos = new FileOutputStream("/home/openkm/import.zip");

            InputStream is = okmNode.exportZip(uuids, true);
            IOUtils.copy(is, fos);
            IOUtils.closeQuietly(is);
            IOUtils.closeQuietly(fos);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNodesFromUuids

Description:

Method	*Return values*	*Description*
getNodesFromUuids(String token, String[] uuids)	***List<Node>***	*Return a list of nodes.*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            String[] uuids = {"0f6463f3-4d36-4091-b518-4fe7c353ee70", "d386cff8-1d4d-
            List<Node> nodes = okmNode.getNodesFromUuids(null, uuids);
            for (Node node : nodes) {
                System.out.println(node);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

evaluateDownloadZip

Description:

Method	*Return values*	*Description*

evaluateDownloadZip(String token, List<String> uuids)	***ZipDownloadEvaluationResult***	*Evaluate when a zip file can be downloaded in real-time or at background.*

> *Related configuration parameters:*
>
> - ***download.zip.live.max.file.size***
> - ***download.zip.live.max.files***
>
> *More information at Performance configuration parameters*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMNode;
import com.openkm.bean.ZipDownloadEvaluationResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            List<String> uuids = new ArrayList<>();
            uuids.add("0f6463f3-4d36-4091-b518-4fe7c353ee70");
            uuids.add("d386cff8-1d4d-472f-9c6d-f21955ec499a");

            ZipDownloadEvaluationResult result = okmNode.evaluateZipDownload(null, uu
            System.out.println(result);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

restore

Description:

Method	*Return values*	*Description*
restore(String token, String nodeId)	***Node***	*Restore a node from the trash.*

> *When a node is moved to trash, the application keeps the origin. When possible the application will try to restore in the same origin (sometimes is not possible because the origin no longer exist or security has been*

changed in the origin, etc.).

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.bean.Node;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            Node node = okmNode.restore(null, "0f6463f3-4d36-4091-b518-4fe7c353ee70")
            System.out.println(node);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

hasNodesLockedByOtherUser

Description:

Method	*Return values*	*Description*
hasNodesLockedByOtherUser(String token, String nodeId)	***booelan***	*Check if the node and descendants have been locked by other users*

This method is used to evaluate if it is possible to lock a hierarchy. For example, in case of a record, the lock affects all the nodes into what also will be locked.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            if (okmNode.hasNodesLockedByOtherUser(null, "0f6463f3-4d36-4091-b518-4fe7
                System.out.println("Is blocked");
            }
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

lock

Description:

Method	*Return values*	*Description*
lock(String token, String nodeId)	***LockInfo***	*Locks a node and returns an object with the Lock information.*

> *Only the user who locked the document is allowed to unlock.*
>
> *A locked document cannot be modified by other users.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.lock(null, "1ec49da9-1746-4875-ae32-9281d7303a62");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forceLock

Description:

Method	*Return values*	*Description*
forceLock(String token, String nodeId)	***void***	*Locks a locked node.*

This method allows to lock any locked node.

> *This action can only be done by users with administrator grants.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.forceLock(null, "1ec49da9-1746-4875-ae32-9281d7303a62");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unlock

Description:

Method	*Return values*	*Description*
unlock(String token, String nodeId)	***void***	*Unlocks a locked node.*

Only the user who locked the document is allowed to unlock.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.unlock(null, "1ec49da9-1746-4875-ae32-9281d7303a62");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forceUnlock

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
forceUnlock(String token, String nodeId)	**void**	*Unlocks a locked node.*

This method allows to unlock any locked node.

It is not mandatory execute this action by the same user who previously executed the checkout lock action.

This action can only be done by a user with administrator grants.

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.forceUnlock(null, "1ec49da9-1746-4875-ae32-9281d7303a62");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

isLocked

Description:

Method	Return values	Description
isLocked(String token, String nodeId)	**Boolean**	*Returns a boolean that indicates if the node is locked or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            System.out.println(okmNode.getLockInfo(null, "1ec49da9-1746-4875-ae32-928
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getLockInfo

Description:

Method	*Return values*	*Description*
getLockInfo(String token, String nodeId)	***LockInfo***	*Returns an object with the Lock information*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            System.out.println(okmNode.getLockInfo(null, "1ec49da9-1746-4875-ae32-928
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setComment

Description:

Method	*Return values*	*Description*
setComment(String token, String nodeId, String versionName, String comment)	***void***	*Sets the comment for a specific node version.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNode;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMNode okmNode = ContextWrapper.getContext().getBean("okmNode", OKMNode.
            okmNode.setComment(null, "1ec49da9-1746-4875-ae32-9281d7303a62", "1.14",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMNote

Basics

*On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID**.*

Example of nodeId:

- *Using UUID -> "**b153c4b7-3d1c-4589-bd42-0ed0f34fd338**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

addNote

Description:

Method	*Return values*	*Description*
add(String token, String nodeId, String text)	***Note***	*Adds a note to a node and returns an object Note.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.Note;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
            Note note = okmNote.add(null, "f123a950-0329-4d62-8328-0ff500fd42db", "th
            System.out.println(note);
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

get

Description:

Method	*Return values*	*Description*
get(String token, String noteId)	***Note***	*Retrieves the note.*

> *The noteId is an UUID.*
>
> *The object Node has a variable named path, in that case the path contains an UUID.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.Note;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
            for (Note note : okmNote.list(null, "f123a950-0329-4d62-8328-0ff500fd42db
                Note data = okmNote.get(null, note.getPath());
                System.out.println(data);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, String noteId)	***void***	*Delete a note.*

> *The noteId is an UUID.*
>
> *The object Node has a variable named path, in that case the path contains an UUID.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.Note;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
            for (Note note : okmNote.list(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338
                okmNote.delete(null, note.getPath());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

set

Description:

Method	*Return values*	*Description*
set(String token, String noteId, String text)	***void***	*Changes the note text.*

> *The noteId is an UUID.*
>
> *The object Node has a variable named path, in that case the path contains an UUID.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.Note;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
            for (Note note : okmNote.list(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338
                okmNote.set(null, note.getPath(), "new note");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

list

Description:

Method	*Return values*	*Description*
list(String token, String nodeId)	***List<Note>***	*Retrieves a list of all the notes of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.Note;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
            for (Note note : okmNote.list(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338
                System.out.println(note);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNoteHistories

Description:

Method	*Return values*	*Description*
getNoteHistories(String token, String nodeId)	***List<NoteHistory>***	*Return the historical notes of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNote;
import com.openkm.bean.NoteHistory;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNote okmNote = ContextWrapper.getContext().getBean(OKMNote.class);
```

```
            for (NoteHistory noteHistory : okmNote.getNoteHistories(null, "3c68b3a1-c
                System.out.println(noteHistory);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMNotification

Basics

*On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID**.*

Example of nodeId:

- *Using uuid -> "**b153c4b7-3d1c-4589-bd42-0ed0f34fd338**"*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

subscribe

Description:

Method	*Return values*	*Description*
subscribe(String token, String nodeId)	***void***	*Adds a subscription to a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNotification;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNotification okmNotification = ContextWrapper.getContext().getBean(OKM

            okmNotification.subscribe(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
}
```

subscribe

Description:

Method	*Return values*	*Description*
unsubscribe(String token, String nodeId)	***void***	*Delete a subscription to a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNotification;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNotification okmNotification = ContextWrapper.getContext().getBean(OKM
            okmNotification.unsubscribe(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338")
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getSubscriptors

Description:

Method	*Return values*	*Description*
getSubscriptors(String token, String nodeId)	***Set<String>***	*Retrieves a list of all the subscriptors of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMNotification;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNotification okmNotification = ContextWrapper.getContext().getBean(OKM
            for (String subscriptor : okmNotification.getSubscriptors(null, "b153c4b7
                System.out.println(subscriptor);
            }
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

notify

Description:

Method	*Return values*	*Description*
notify(String token, String nodeId, List<String> users, List<String> mails, String message, boolean attachment)	***void***	*Send a mail notification.*

The parameter users are a set of OpenKM users to be notified.

The parameter mails are a set of mails - usually external mails - to be notified.

The parameter message is the content body of the mail.

When attachment value is true the node is attached into the mail.

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMNotification;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMNotification okmNotification = ContextWrapper.getContext().getBean(OKM
            List<String> users = new ArrayList<>();
            users.add("sochoa");
            List<String> mails = new ArrayList<>();
            String body = "Body of the message";
            okmNotification.notify(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", user
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMProperty

Basics

*On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID**.*

Example of nodeId:

- *Using UUID -> "**b153c4b7-3d1c-4589-bd42-0ed0f34fd338**";*

*About the parameter named "**catId**", the value of this parameter can be a valid categories folder **UUID** node.*

Example of nodeId:

- *Using UUID -> "**5cbc35fc-23c0-48f5-a7bc-3cfa1e7be25f**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

addCategory

Description:

Method	*Return values*	*Description*
addCategory(String token, String nodeId, String catId)	***void***	*Set a relation between a category and a node.*
The value of the catId parameter should be a category folder UUID.		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.addCategory(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "5c
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

removeCategory

Description:

Method	*Return values*	*Description*
removeCategory(String token, String nodeId, String catId)	***void***	*Removes a relation between a category and a node.*
The value of the catId parameter should be a category folder UUID.		

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.removeCategory(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addKeyword

Description:

Method	*Return values*	*Description*

addKeyword(String token, String nodeId, String keyword)	***String***	*Add a keyword in a node.*

The keyword should be a single word without spaces, formats allowed:

- *"test"*
- *"two_words" (the character "_" is used for the junction).*

We also we suggest you to add keyword in lowercase format, because OpenKM is case sensitive.

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            String keyword = okmProperty.addKeyword(null, "b153c4b7-3d1c-4589-bd42-0e
            System.out.println(keyword);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

removeKeyword

Description:

Method	*Return values*	*Description*
removeKeyword(String token, String nodeId, String keyword)	***void***	*Removes a keyword from a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.removeKeyword(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setEncryption

Description:

Method	*Return values*	*Description*
setEncryption(String token, String nodeId, String cipherName)	***void***	*Marks a document as an en cripted binary data in the repository.*

The parameter nodeId should be a document node.

The parameter cipherName saves information about the encription mechanism.

This method does not perform any kind of encryption, simply mark in the database that a document is encrypted.

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.setEncryption(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unsetEncryption

Description:

Method	*Return values*	*Description*
unsetEncryption(String token, String	***void***	*Marks a document is a normal binary data into*

nodeId)		*repository.*

The parameter nodeId should be a document node.

This method does not perform any kind of encryption, simply mark into the database that a document has been uncrypted.

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.unsetEncryption(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338")
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setSigned

Description:

Method	*Return values*	*Description*
setSigned(String token, String nodeId, boolean signed)	***void***	*Marks a document as signed or unsigned binary data into the repository.*

The parameter nodeId should be a document node.

This method does not perform any kind of digital signature process, simply mark into the database that a document is signed.

Example:

```
package com.openkm;

import com.openkm.api.OKMProperty;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMProperty okmProperty = ContextWrapper.getContext().getBean(OKMProperty
            okmProperty.setSigned(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", true)
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMPropertyGroup

Basics

*From the older OpenKM version we named "**Metadata Groups**" as "**Property Groups**".*

Although we understand this name does not help a lot to identify these methods with metadata ones, for historical reason, we continue maintaining the nomenclature.

For more information about Metadata.

*On most methods you'll see a parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID**.*

Example of nodeId:

- *Using UUID -> "**b153c4b7-3d1c-4589-bd42-0ed0f34fd338**";*

The class com.openkm.util.ISO8601 should be used to set and parse metadata date fields. The metadata field of type date values is stored in the application in ISO-8601 basic format.

To convert the retrieved metada field of type date to a valid date use:

```
Calendar cal = ISO8601.parseBasic(metadataFieldValue);
```

To save the date value in the metadata field of type date use:

```
Calendar cal = Calendar.getInstance(); // Present date
String metadataFieldValue = ISO8601.formatBasic(cal);
// metadataFieldValue can be saved into repository metadata field of type dat
```

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

addGroup

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Descript*
addGroup(String token, String nodeId, String grpName, Map<String, String> properties)	***void***	*Adds an metadata group node.*

The grpName should be a valid Metadata group name.

It is not mandatory to set in the properties parameter all fields values, is enough with the fields you wish to change

The sample below is based on this Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"
                                 "http://www.openkm.com/dtd/property-groups
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Name" type="text" name="okp:consulting.name"/>
    <input label="Date" type="date" name="okp:consulting.date" />
    <checkbox label="Important" name="okp:consulting.important"/>
    <textarea label="Comment" name="okp:consulting.comment"/>
  </property-group>
</property-groups>
```

To add several values on a metadata field of type multiple like this:

```
<select label="Multiple" name="okp:consulting.multiple" type="multiple">
  <option label="One" value="one"/>
  <option label="Two" value="two"/>
  <option label="Three" value="three" />
</select>
```

You should do:

```
properties.put("okp:consulting.multiple", "one;two");
```

*Where "**one**" and "**two**" are valid values and character "**;**" is used as separator.*

Example:

```
package com.openkm;

import java.util.Calendar;
import java.util.HashMap;
```

```
import java.util.Map;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;
import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            Map<String, String> properties = new HashMap<>();
            properties.put("okp:consulting.name", "new name");
            properties.put("okp:consulting.date", ISO8601.formatBasic(Calendar.getIns
            properties.put("okp:consulting.important", String.valueOf(true));
            properties.put("okp:consulting.comment", "new comment");

            okmPropertyGroup.addGroup(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338", "
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

removeGroup

Description:

Method	*Return values*	*Description*
removeGroup(String token, String nodeId, String grpName)	***void***	*Removes a metadata group of a node.*
The grpName should be a valid Metadata group name.		

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            okmPropertyGroup.removeGroup(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd338"
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getGroups

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
getGroups(String token, String nodeId)	***List<PropertyGroup>***	*Retrieves a list of metadata groups assigned to a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;
import com.openkm.bean.PropertyGroup;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            for (PropertyGroup pg : okmPropertyGroup.getGroups(null, "b153c4b7-3d1c-4
                System.out.println(pg);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAllGroups

Description:

Method	Return values	Description
getAllGroups(String token)	***List<PropertyGroup>***	*Retrieves a list of all metadata groups set into the application.*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;
import com.openkm.bean.PropertyGroup;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            for (PropertyGroup pg : okmPropertyGroup.getAllGroups(null)) {
                System.out.println(pg);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String nodeId, String grpName)	***Map<String, String>***	*Retrieves a list of all metadata group elements and its node values.*

The grpName should be a valid Metadata group name.

> *The method is usually used to display form elements with its values to be shown or changed by used.*

Example:

```
package com.openkm;

import java.util.Map;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            Map<String, String> properties = okmPropertyGroup.getProperties(null, "b1
            for (String key : properties.keySet()) {
                System.out.println(key + ": " + properties.get(key));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setProperties

Description:

Method	*Return values*	*Descri*
setProperties(String token, String nodeId, String grpName, Map<String, String> properties)	***void***	*Changes the metadata node.*

The grpName should be a valid Metadata group name.

It is not mandatory to set in the properties parameter all fields values, is enough with the fields you wish to change

The sample below is based on this Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"
                                 "http://www.openkm.com/dtd/property-groups
<property-groups>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Name" type="text" name="okp:consulting.name"/>
    <input label="Date" type="date" name="okp:consulting.date" />
    <checkbox label="Important" name="okp:consulting.important"/>
    <textarea label="Comment" name="okp:consulting.comment"/>
  </property-group>
</property-groups>
```

To add several values on a metadata field of type multiple like this:

```
<select label="Multiple" name="okp:consulting.multiple" type="multiple">
  <option label="One" value="one"/>
  <option label="Two" value="two"/>
  <option label="Three" value="three" />
</select>
```

You should do:

```
properties.put("okp:consulting.multiple", "one;two");
```

*Where "**one**" and "**two**" are valid values and character "**;**" is used as separator.*

Example:

```
package com.openkm;

import java.util.Calendar;
import java.util.HashMap;
import java.util.Map;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;
import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            Map<String, String> properties = new HashMap<>();
```

```
            properties.put("okp:consulting.name", "new name");
            properties.put("okp:consulting.date", ISO8601.formatBasic(Calendar.getIns
            properties.put("okp:consulting.important", String.valueOf(true));
            properties.put("okp:consulting.comment", "new comment");

            okmPropertyGroup.setProperties(null, "b153c4b7-3d1c-4589-bd42-0ed0f34fd33
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPropertyGroupForm

Description:

Method	*Return values*	*Description*
getPropertyGroupForm(String token, String grpName)	***List<FormElement>***	*Retrieves a list of all metadata group elements definition.*

The grpName should be a valid Metadata group name.

> *The method is usually used to display empty form elements for creating new metadata values.*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.bean.form.FormElement;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            for (FormElement fe : okmPropertyGroup.getPropertyGroupForm(null, "okg:co
                System.out.println(fe);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

hasGroup

Description:

Method	*Return*	*Description*

values		
hasGroup(String token, String nodeId, String grpName)	***boolean***	*Returns a boolean that indicate if the node has or not a metadata group.*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            boolean found = okmPropertyGroup.hasGroup(null, "b153c4b7-3d1c-4589-bd42-
            System.out.println(found);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getSuggestions

Description:

Method	Return values	Descriptio
getSuggestions(String token, String nodeId, String grpName, String propName)	***List<String>***	*Retrieves a list of a suggestec values.*

The propName parameter should be a Metadata Select field type.

More information at Creating your own Suggestion Analyzer and Metadata Select field .

The sample below is based on this Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.0//EN"
                                 "http://www.openkm.com/dtd/property-groups
<property-groups>
  <property-group label="Technology" name="okg:technology">
    <select label="Type" name="okp:technology.type" type="multiple">
      <option label="Alfa" value="t1"/>
```

```
        <option label="Beta" value="t2" />
        <option label="Omega" value="t3" />
      </select>
      <select label="Language" name="okp:technology.language" type="simple">
        <option label="Java" value="java"/>
        <option label="Python" value="python"/>
        <option label="PHP" value="php" />
      </select>
      <input label="Comment" name="okp:technology.comment"/>
      <textarea label="Description" name="okp:technology.description"/>
      <input label="Link" type="link" name="okp:technology.link"/>
    </property-group>
  </property-groups>
```

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            for (String value : okmPropertyGroup.getSuggestions(null, "b153c4b7-3d1c-
                    "okp:technology.language")) {
                System.out.println(value);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

registerDefinition

Description:

Method	*Return values*	*Description*
registerDefinition(String token, String pgDef, String pgName)	***void***	*Set the XML Metada groups definition into the repository.*

The method only can be executed by super user granted (ROLE_ADMIN member user).

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        InputStream is = null;
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            is = new FileInputStream("/home/files/PropertyGroups.xml");
            String pgDef = IOUtils.toString(is);
            okmPropertyGroup.registerDefinition(null, pgDef, "test");
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(is);
        }
    }
}
```

getRegisteredDefinition

Description:

Method	*Return values*	*Description*
getRegisteredDefinition(String token)	***String***	*Returns the XML Metada groups definition.*

The method only can be executed by super user granted (ROLE_ADMIN member user).

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            String pgDef = okmPropertyGroup.getRegisteredDefinition(null);
            System.out.println(pgDef);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

getGroups

Description:

Method	*Return values*	*Description*
getGroups(String token, String nodeId, String versionName)	***List<PropertyGroup>***	*Retrieves a list of metadata groups assigned to a specific version of a node*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.bean.PropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            for (PropertyGroup pg : okmPropertyGroup.getGroups(null, "82555dd1-bcc2-4
                System.out.println(pg);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getGroupProperties

Description:

Method	*Return values*	*Description*
getGroupProperties(String token, Long pgdId)	***List<PropertyGroup>***	*Retrieves a list of metadata groups properties.*

Example:

```
package com.openkm;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.bean.PropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O
            long pgdId = 1;
            for (PropertyGroup pg : okmPropertyGroup.getGroupProperties(null, pgdId))
                System.out.println(pg);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String nodeId, String grpName, Sring versionName)	***Map<String, String>***	*Retrieves a map - (key, value) pairs - with Metadata group values assigned to a specific version of a node.*

Example:

```
package com.openkm;

import java.util.Map;

import com.openkm.api.OKMPropertyGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMPropertyGroup okmPropertyGroup = ContextWrapper.getContext().getBean(O

            Map<String, String> properties = okmPropertyGroup.getProperties(null, "82
            for (String key : properties.keySet()) {
                System.out.println(key + ": " + properties.get(key));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMRecord

Basics

*On almost methods you'll see parameter named "**recId**". The value of this parameter can be some valid record **UUID**.*

Example of recId:

- *Using UUID -> "**7f867772-7476-4176-b86f-bdeb8bce1c5f**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

create

Description:

Method	*Return values*	*Description*
create(String token, Record rec)	***Record***	*Creates a new record and returns as a result an object Record.*

*The variable **path** into the parameter **rec**, must be initialized. It indicates the folder path into OpenKM.*

```
Record rec = new Record();
rec.setPath("/okm:root/PKI-100200");
```

The other variables of the Record (rec) will not take any effect on the record creation.

We suggest using the method below to create the Record Simply rather this one.

Example:

```
package com.openkm;
```

```
import com.openkm.api.OKMRecord;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            Record rec = new Record();
            rec.setPath("/okm:root/test");
            okmRecord.create(null, rec);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createSimple

Description:

Method	*Return values*	*Description*
createSimple(String token, String recPath)	***Record***	*Creates a new record and returns as a result an object Record.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            String recPath = "/okm:root/PKI-100200";
            okmRecord.createSimple(null, recPath);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProperties

Description:

Method	*Return values*	*Description*
getProperties(String token, String recId)	***Record***	*Return the record properties.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            Record rec = okmRecord.getProperties(null, "7f867772-7476-4176-b86f-bdeb8
            System.out.println(rec);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, String recId)	***void***	*Delete a record.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.delete(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

purge

Description:

Method	*Return values*	*Description*

purge(String token, String recId)	***void***	*The record is definitely removed from the repository.*

Usually you will purge folders into /okm:trash/userId - the personal trash user locations - but it is possible to directly purge any record from the whole repository.

When a record is purged, it will only be able to be restored from a previously repository backup. The purge action removes the record definitely from the repository.

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.purge(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

rename

Description:

Method	*Return values*	*Description*
rename(String token, String recId, String newName)	***Record***	*Rename a record.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.rename(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "newname")
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

move

Description:

Method	*Return values*	*Description*
move(String token, String recId, String dstId)	***void***	*Moves record into some folder or record*
The values of the dstId parameter should be a folder or record UUID.		

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.move(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "b6848ac2-34
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String recId, String dstId)	***void***	*Copies a record into a folder or record.*

The values of the dstId parameter should be a folder or record UUID.

Only the binary data and the security grants are copied to destination, the metadata, keywords, etc. of the document are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.copy(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "b6848ac2-34
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

copy

Description:

Method	*Return values*	*Description*
copy(String token, String recId, String dstId, String newName)	***void***	*Copies a record into a folder or record.*

The values of the dstId parameter should be a folder or record UUID.

When parameter newName value is null, folder will preservate the same name.

Only the binary data and the security grants are copied to destination, the metadata, keywords, etc. of the document are not copied.

*See "**extendedCopy**" method for this feature.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.copy(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "b6848ac2-34
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

extendedCopy

Description:

Method	*Return values*	*Description*
extendedCopy(String token, String recId, String dstId, String newName, ExtendedAttributes extAttr)	***void***	*Copies a record with associated data into some folder or record.*

The values of the dstId parameter should be a folder or a record UUID.

When the parameter newName value is null, the record will preserve the same name.

By default only the binary data and the security grants, the metadata, keywords, etc. of the document are not copied.

Additional:

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the propertyGroups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original value of the notes will be copied.*
- *When wiki parameter is true the original value of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setKeywords(true);
            extAttr.setCategories(true);
            extAttr.setNotes(true);

            okmRecord.extendedCopy(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "b67
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

isValid

Description:

Method	*Return values*	*Description*
isValid(String token, String recId)	***boolean***	*Returns a boolean that indicates if the node is a record or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            boolean valid = okmRecord.isValid(null, "7f867772-7476-4176-b86f-bdeb8bce
            System.out.println(valid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPath

Description:

Method	*Return values*	*Description*
getPath(String token, String recId)	***String***	*Convert folder UUID to record path.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
            String path = okmRecord.getPath(null, "7f867772-7476-4176-b86f-bdeb8bce1c
            System.out.println(path);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChildren

Description:

Method	*Return values*	*Description*
getChildren(String token, String fldId)	***List<Record>***	*Returns a list of all records which their parent is fldId.*
The parameter fldId can be a folder or a record node.		

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            for (Record rec : okmRecord.getChildren(null, "b6848ac2-345f-454e-964a-eb
                System.out.println(rec);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

lock

Description:

Method	*Return values*	*Description*
lock(String token, String recId)	***LockInfo***	*Locks a record and return an object with the Lock information.*

Only the user who locked the record is allowed to unlock.

A locked record can not be modified by other users.

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.lock(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

unlock

Description:

Method	*Return values*	*Description*
unlock(String token, String recId)	***LockInfo***	*Unlock a locked record.*

Only the user who locked the record is allowed to unlock.

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.unlock(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

forceUnlock

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
forceUnlock(String token, String recId)	**void**	*Unlock a locked record.*

This method allows to unlock any locked record.

It is not mandatory to execute this action by the same user who previously executed the checkout lock action.

This action can only be done by a super user (user with ROLE_ADMIN).

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
            okmRecord.forceUnlock(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTitle

Description:

Method	Return values	Description
setTitle(String token, String recId, String title)	**void**	*Sets a record title.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class
```

```
            okmRecord.setTitle(null, "7f867772-7476-4176-b86f-bdeb8bce1c5f", "new tit
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setNodeClass

Description:

Method	*Return values*	*Description*
setNodeClass(String token, String recId, long ncId)	***void***	*Set the NodeClass.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class

            long ncId = 2;
            okmRecord.setNodeClass(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b", ncId
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setDispositionStage

Description:

Method	*Return values*	*Description*
setDispositionStage(String token, String recId, long stage)	***void***	*Set the disposition stage*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class

            long stage = 1;
            okmRecord.setDispositionStage(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addNodeClassChildren

Description:

Method	*Return values*	*Description*
addNodeClassChildren(String token, String recId, long ncId)	***void***	*Add NodeClass of the record in the folder parents.*
Used for navigation purposes to evaluating when a folder contains a node of type node class. *The method propagate the node class value in the parent containers (folders) to indicate there's a node of this type.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class

            long ncId = 2;
            okmRecord.addNodeClassChildren(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setDescription

Description:

Method	*Return values*	*Description*

setDescription(String token, String uuid, String description)	***void***	*Set a description.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class

            okmRecord.setDescription(null, "7ce1b4a8-4ade-4dce-8d7d-4e99a6cd368b", "s
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createWizardRecord

Description:

Method	*Return values*	*Description*
createWizard(String token, String recPath, String title, long nodeClass)	***WizardNode***	*Create a new record with wizard.*

The parameters ***recPath*** *should be any valid folder or record* ***UUID***

The WizardNode contains a list of pending actions what should be done to complete the process of record creation. These might be:

- *Add keyword*
- *Add Categories*
- *Add Metadata*

Example:

```
package com.openkm;

import com.openkm.api.OKMRecord;
import com.openkm.bean.WizardNode;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMRecord okmRecord = ContextWrapper.getContext().getBean(OKMRecord.class

            WizardNode wn = okmRecord.createWizard(null, "1f323e88-64ee-4f57-91e2-927
            System.out.print(wn);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMRelation

Basics

*On most methods you'll see parameter named "**nodeId**". The value of this parameter can be a valid document, folder, mail or record **UUID**.*

Example of nodeId:

- *Using UUID -> "**d168924d-e801-426d-a222-00124a1461bd**";*

*Also on all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken()
```

Methods

getRelationTypes

Description:

Method	*Return values*	*Description*
getRelationTypes(String token, String type)	***List<RelationType>***	*Retrieves a list of all relations defined of a type.*

Available types values:

- *NodeRelationType.BIDIRECTIONAL*
- *NodeRelationType.PARENT_CHILD*
- *NodeRelationType.MANY_TO_MANY*

More information at Relation types .

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationType;
import com.openkm.db.bean.NodeRelationType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationType rt : okmRelation.getRelationTypes(null, NodeRelationTyp
                System.out.println(rt);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

add

Description:

Method	*Return values*	*Description*
add(String token, String nodeAId, String nodeBId, long relationType)	***void***	*Sets a relation between two nodes.*
The parameters ***nodeAId*** *and* ***nodeBId*** *should be any valid document, folder, mail or record* ***UUID****.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationType;
import com.openkm.db.bean.NodeRelationType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationType rt : okmRelation.getRelationTypes(null, NodeRelationTyp
                // looking for a relation named invoice
                if (rt.getTitle().equals("invoice")) {
                    okmRelation.add(null, "d168924d-e801-426d-a222-00124a1461bd", "1f
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, long relationId)	***void***	*Deletes a relation.*

Only when the relation will not be used by any node is able to be deleted, otherwise you'll get an error.

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.Relation;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (Relation relation : okmRelation.getRelations(null, "d168924d-e801-42
                if (relation.getRelationTitle().equals("invoice")) {
                    okmRelation.delete(null, relation.getId());
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRelations

Description:

Method	*Return values*	*Description*
getRelations(String token, String nodeId)	***List<Relation>***	*Retrieves a list of all relations of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
```

```
import com.openkm.bean.Relation;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (Relation relation : okmRelation.getRelations(null, "d168924d-e801-42
                System.out.println(relation);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRelationGroups

Description:

Method	*Return values*	*Description*
getRelationGroups(String token, String nodeId)	***List<RelationGroup>***	*Retrieves a list of all relation groups of a node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationGroup rg : okmRelation.getRelationGroups(null, "d168924d-e80
                System.out.println(rg);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addGroup

Description:

Method	*Return values*	*Description*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

addGroup(String token, String nodeId, String groupName, long type)	***void***	*Adds a relation group at a node.*

On a relation group only has sense to apply a relation type of NodeRelationType.MANY_TO_MANY.

More information at Relation types .

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationType;
import com.openkm.db.bean.NodeRelationType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationType rt : okmRelation.getRelationTypes(null, NodeRelationTyp
                if (rt.getTitle().equals("staple")) {
                    okmRelation.addGroup(null, "d168924d-e801-426d-a222-00124a1461bd"
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addNodeToGroup

Description:

Method	*Return values*	*Description*
addNodeToGroup(String token, String nodeId, long groupId)	***void***	*Adds a node to an existing relation group.*

On a relation group only has sense to apply a relation type of NodeRelationType.MANY_TO_MANY.

More information at Relation types .

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationGroup rg : okmRelation.getRelationGroups(null, "d168924d-e80
                if (rg.getRelationTitle().equals("staple")) {
                    okmRelation.deleteGroup(null, "d168924d-e801-426d-a222-00124a1461
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteGroup

Description:

Method	*Return values*	*Description*
deleteGroup(String token, String nodeId, long groupId)	***void***	*Remove a node from a relation group.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            for (RelationGroup rg : okmRelation.getRelationGroups(null, "8f101a85-88e
                if (rg.getName().equals("staple group")) {
                    okmRelation.deleteGroup(null, "46762f90-82c6-4886-8d21-ad3017dd78
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getGroup

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	Return values	Description
getGroup(String token, long groupId)	**RelationGroup**	*Finds a relation group by id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.bean.RelationGroup;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            long groupId = 1;
            RelationGroup rg = okmRelation.getGroup(null, groupId);
            System.out.println(rg);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setGroupName

Description:

Method	Return values	Description
setGroupName(String token, long groupId, String groupName)	**void**	*Changes the relation group name.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRelation;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRelation okmRelation = ContextWrapper.getContext().getBean(OKMRelation
            long groupId = 1;
            okmRelation.setGroupName(null, groupId, "new name");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMReport

Basics

You can set several reports formats based in document mime type:

Contants defined into Report class:

- *Report.FORMAT_TEXT = "text/plain"*
- *Report.FORMAT_HTML = "text/html"*
- *Report.FORMAT_CSV = "text/csv"*
- *Report.FORMAT_PDF = "application/pdf"*
- *Report.FORMAT_RTF = "application/rtf"*
- *Report.FORMAT_ODT = "application/vnd.oasis.opendocument.text"*
- *Report.FORMAT_XLSX = "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet"*
- *Report.FORMAT_DOCX = "application/vnd.openxmlformats-officedocument.wordprocessingml.document"*

Methods

list

Description:

Method	*Return values*	*Description*
list(String token, boolean active)	***List<Report>***	*Returns a list of reports.*
*When variable **active** when is set to **true** will be returned only the active reports otherwise all the reports.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMReport;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMReport okmReport = ContextWrapper.getContext().getBean(OKMReport.class
            System.out.println(okmReport.list(null, true));
        } catch (Exception e) {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
            e.printStackTrace();
        }
    }
}
```

get

Description:

Method	*Return values*	*Description*
get(String token, long rpId)	***Report***	*Returns an object Report.*
*The variable **rpId** is the long value used to identify the report.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMReport;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMReport okmReport = ContextWrapper.getContext().getBean(OKMReport.class
            System.out.println(okmReport.get(null, 1));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

execute

Description:

Method	*Return values*	*Description*
execute(String token, long rpId, Map<String, Object> params, String format)	***InputStream***	*Returns the inputstream of the report.*
*The variable **rpId** is the long value used to identify the report.*		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import java.io.InputStream;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.HashMap;
import java.util.Map;

import com.openkm.api.OKMReport;
import com.openkm.bean.Report;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMReport okmReport = ContextWrapper.getContext().getBean(OKMReport.class

            Calendar yesterday = Calendar.getInstance();
            yesterday.add(Calendar.DATE, -1);
            Calendar today = Calendar.getInstance();

            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

            Map<String, Object> params = new HashMap<>(); // report parameters
            params.put("from_date", sdf.format(yesterday.getTime()));
            params.put("to_date", sdf.format(today.getTime()));
            InputStream is = okmReport.execute(null, 1, params, Report.FORMAT_PDF);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMRepository

*On all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

getRootFolder

Description:

Method	*Return values*	*Description*
getRootFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:root".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getRootFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getTrashFolder

Description:

Method	*Return values*	*Description*
getTrashFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:trash/{userId}".*

The returned folder will be the user trash folder.

> *For example if the method is executed by the user "okmAdmin" then the folder returned will be "/okm:trash/okmAdmin".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getTrashFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getTrashFolderBase

Description:

Method	*Return values*	*Description*
getTrashFolderBase(String token)	***Folder***	*Returns the object Folder of node "/okm:trash".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getTrashFolderBase(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
    }
}
```

getTemplatesFolder

Description:

Method	*Return values*	*Description*
getTemplatesFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:templates".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getTemplatesFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPersonalFolder

Description:

Method	*Return values*	*Description*
getPersonalFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:personal/{userId}".*

The returned folder will be the user personal folder.

> *For example if the method is executed by the user "okmAdmin" then the folder returned will be "/okm:personal/okmAdmin".*

Example:

```
package com.openkm;
```

```
import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getPersonalFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPersonalFolderBase

Description:

Method	*Return values*	*Description*
getPersonalFolderBase(String token)	***Folder***	*Returns the object Folder of node "/okm:personal".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getPersonalFolderBase(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMailFolder

Description:

Method	*Return values*	*Description*
getMailFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:mail/{userId}".*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The returned folder will be the user mail folder.

> *For example if the method is executed by the user "okmAdmin" then the folder returned will be "/okm:mail/okmAdmin".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getMailFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMailFolderBase

Description:

Method	*Return values*	*Description*
getMailFolderBase(String token)	***Folder***	*Returns the object Folder of node "/okm:mail".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getMailFolderBase(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getThesaurusFolder

Description:

Method	*Return values*	*Description*
getThesaurusFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:thesaurus".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getThesaurusFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getCategoriesFolder

Description:

Method	*Return values*	*Description*
getCategoriesFolder(String token)	***Folder***	*Returns the object Folder of node "/okm:categories".*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            Folder fld = okmRepository.getCategoriesFolder(null);
            System.out.println(fld);
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

purgeTrash

Description:

Method	*Return values*	*Description*
purgeTrash(String token)	***Folder***	*Definitively removes from repository all nodes to "/okm:trash/{userId}".*

For example if the method is executed by the user "okmAdmin" then the purged trash will be "/okm:trash/okmAdmin".

When a node is purged it will only be able to be restored from a previously repository backup. The purge action removes the node definitely from the repository.

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            okmRepository.purgeTrash(null);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getUpdateMessage

Description:

Method	*Return values*	*Description*
getUpdateMessage(String token)	***String***	*Retrieves a message when there is a new OpenKM release.*

There's an official OpenKM update message service available which is based on your local OpenKM version.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            String msg = okmRepository.getUpdateMessage(null);
            System.out.println(msg);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRepositoryUuid

Description:

Method	*Return values*	*Description*
getRepositoryUuid(String token)	***String***	*Retrieves the installation unique identifier.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            String uuid = okmRepository.getRepositoryUuid(null);
            System.out.println(uuid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

hasNode

Description:

Method	Return values	Description
hasNode(String token, String nodeId)	**boolean**	Returns a node that indicate if a node exists or not.
The value of the parameter nodeId can be a valid UUID.		

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            boolean found = okmRepository.hasNode(null, "b67fe3ae-bcca-4947-b8ea-0cc7
            System.out.println(found);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNodePath

Description:

Method	Return values	Description
getNodePath(String token, String uuid)	**String**	Converts a node UUID to path.

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            String path = okmRepository.getNodePath(null, "b67fe3ae-bcca-4947-b8ea-0c
            System.out.println(path);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
}
```

getNodeUuid

Description:

Method	*Return values*	*Description*
getNodeUuid(String token, String path)	***String***	*Converts a node path to UUID.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            String uuid = okmRepository.getNodeUuid(null, "/okm:root/api");
            System.out.println(uuid);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAppVersion

Description:

Method	*Return values*	*Description*
getAppVersion(String token)	***AppVersion***	*Returns information about OpenKM version.*

Example:

```
package com.openkm;
import com.openkm.api.OKMRepository;
import com.openkm.bean.AppVersion;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            AppVersion appVersion = okmRepository.getAppVersion(null);
            System.out.println(appVersion);
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

copyAttributes

Description:

Method	*Return values*	*Description*
copyAttributes(String token, String srcId, String dstId, ExtendedAttributes extAttr)	***void***	*Copy attributes from a node to other..*

The values of the dstId parameter should be a node ***UUID****.*

- *When the category parameter is true the original values of the categories will be copied.*
- *When the keywords parameter is true the original values of the keywords will be copied.*
- *When the property Groups parameter is true the original values of the metadata groups will be copied.*
- *When the notes parameter is true the original values of the notes will be copied.*
- *When the wiki parameter is true the original values of the wiki will be copied.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.ExtendedAttributes;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo
            ExtendedAttributes extAttr = new ExtendedAttributes();
            extAttr.setKeywords(true);
            extAttr.setCategories(true);
            extAttr.setNotes(true);
            extAttr.setPropertyGroups(true);
            extAttr.setWiki(true);
            okmRepository.copyAttributes(null, "d168924d-e801-426d-a222-00124a1461bd"
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMetadataFolder

Description:

Method	*Return values*	*Description*
getMetadataFolder(String token)	***Folder***	*Return the metadata folder.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo

            Folder folder = okmRepository.getMetadataFolder(null);
            System.out.println(folder);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getChangeLog

Description:

Method	*Return values*	*Description*
getChangeLog(String token, String nodePath, Calendar modifyFrom)	***List<ChangeLogged>***	*Return the list of changes in some path and subfolders.*

- *The method is used by desktop synchronization application for retrieving the changes.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMRepository;
```

```
import com.openkm.bean.ChangeLogged;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo

            Calendar now = Calendar.getInstance();
            for (ChangeLogged cl : okmRepository.getChangeLog(null, "/okm:root/synchr
                System.out.println(cl);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAppLicense

Description:

Method	*Return values*	*Description*
getAppLicense(String token)	***LicenseInfo***	*Return license information.*

Example:

```
package com.openkm;

import com.openkm.api.OKMRepository;
import com.openkm.util.ContextWrapper;
import com.openkm.util.Update.LicenseInfo;

public class Test {

    public static void main(String[] args) {
        try {
            OKMRepository okmRepository = ContextWrapper.getContext().getBean(OKMRepo

            LicenseInfo licenseInfo = okmRepository.getAppLicense(null);
            System.out.println(licenseInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMSearch

Basics

Almosts all methods use QueryParams. Here there're some tips about how using it.

Variables	*Type*	*Allow wildcards*	*Restric*
domain	***long***	*No.*	*Available values:* • *QueryParams.DOCUMENT* • *QueryParams.FOLDER* • *QueryParams.MAIL* • *QueryParams.RECORD* *By default the value is set to QueryParams.DOCUMENT.* *For searching documents and folders use value:* `(QueryParams.DOCUMENT | QueryParams.FOLDER)`
author	***String***	*No.*	*The value must be a valid userId.*
name	***String***	*Yes.*	
title	***String***	*Yes.*	
keywords	***Set<String>***	*Yes.*	
categories	***Set<String>***	*No.*	*Values should be a category UUID, not use path value.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

content		*Yes.*	
mimeType		*No.*	*The value should be a valid and registered MIME type.* *Only can be applied to documents node.*
language		*No.*	*The value should be a valid language.* *Only can be applied to documents node.*
folder		*No.*	*When empty is used by default "/okm:root" node.* *The value should be a valid UUID, not use a path value.*
folderRecursive	***Boolean***	*No.*	*Only has sense to set this variable to true when the variable folder is not*
lastModifiedFrom	***Calendar***	*No.*	
lastModifiedTo	***Calendar***	*No.*	
mailSubject	***String***	*Yes.*	*Only apply to mail nodes.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

mailFrom	***String***	*Yes.*	*Only apply to mail nodes.*
mailTo		*Yes.*	*Only apply to mail nodes.*
notes		*Yes.*	

properties	***Map<String, String>***	*Yes on almost.*	*On metadata field values like "date" can not be applied wilcards.*
			
			The map of the properties is composed of pairs:
			
			('metadata_field_name','metada_field_value")
			
			For example:
			
			`Map<String, String> properties = new HashMap();`
			`properties.put("okp:consulting.name", "name val`
			
			Filtering by range of dates:
			
			`Calendar to = Calendar.getInstance(); // today`
			`to.set(0, Calendar.HOUR);`
			`to.set(0, Calendar.MINUTE);`
			`to.set(0, Calendar.SECOND);`
			`to.set(0, Calendar.MILLISECOND);`
			`Calendar from = (Calendar) to.clone();`
			`from.add(-3, Calendar.DATE); // three days befo`
			`Map<String,String> properties = new HashMap<>()`
			`properties.put("okp:consulting.date", ISO8601.f`
			
			When filtering by a range of dates you must set both values (OpenKM side.
			
			To filtering by a metadata field of type multiple like this:

```
<select label="Multiple" name="okp:consulting.m
  <option label="One" value="one"/>
  <option label="Two" value="two"/>
  <option label="Three" value="three" />
</select>
```

You should do:

```
properties.put("okp:consulting.multiple", "one;
```

*Where "**one**" and "**two**" are valid values and character "**;**" is used as sep*

The search operation is done only by AND logic.

Wildcard examples:

Variable	*Example*	*Description*
name	***test*.html***	*Any document that starts with characters "test" and ends with characters ".html"*
name	***test?.html***	*Any document that starts with the characters "test" followed by a single character and ends with characters ".html"*
name	***?test****	*Any of the documents where the first character doesn't matter but is followed by the characters, "test".*

Methods

find

Description:

Method	*Return values*	*Description*
find(String token, QueryParams params)	***List<QueryResult>***	*Returns a list of results filtered by the values of the queryParams parameter.*

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");

            for (QueryResult qr : okmSearch.find(null, qParams)) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

find

Description:

Method	*Return values*	*Description*
find(String token, QueryParams params, String propertiesPlugin)	***List<QueryResult>***	*Returns a list of results filtered by the values of the queryParams parameter.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only a few Objects variables. If it's your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");

            for (QueryResult qr : okmSearch.find(null, qParams, "com.openkm.plugin.pr
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findPaginated

Description:

Method	*Return values*	*Description*
findPaginated(String token, QueryParams params, int offset, int limit)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");
            ResultSet rs = okmSearch.findPaginated(null, qParams, 20, 10);
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findPaginated

Description:

Method	*Return values*	*Description*
findPaginated(String token, QueryParams params, int offset, int limit, String propertiesPlugin)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*
- *The parameter "propertiesPlugin" must be the canonical class name of the class which implements the NodeProperties interface.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");
            ResultSet rs = okmSearch.findPaginated(null, qParams, 20, 10, "com.openkm
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQuery

Description:

Method	*Return values*	*Description*
findByQuery(String token, String query)	***List<QueryResult>***	*Returns a list of paginated results filtered by the value of the query parameter.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (QueryResult qr : okmSearch.findByQuery(null, "text:grial AND name:o*
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQuery

Description:

Method	*Return values*	*Description*
findByQuery(String token, String query, String propertiesPlugin)	***ResultSet***	*Returns a list of paginated results filtered by the value of the query parameter.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (QueryResult qr : okmSearch.findByQuery(null, "text:grial AND name:o*
                    "com.openkm.plugin.properties.SimpleNodeProperties")) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQueryPaginated

Description:

Method	*Return values*	*Description*

findByQueryPaginated(String token, String query, int offset, int limit)	***ResultSet***	*Returns a list of paginated results filtered by the value of the query parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            ResultSet rs = okmSearch.findByQueryPaginated(null, "text:grial AND name:
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQueryPaginated

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	*Return values*	*Description*
findByQueryPaginated(String token, String query, int offset, int limit, String propertiesPlugin)	***ResultSet***	*Returns a list of paginated results filtered by the value of the query parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            ResultSet rs = okmSearch.findByQueryPaginated(null, "text:grial AND name:
                    "com.openkm.plugin.properties.SimpleNodeProperties");
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
```

```
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

saveSearch

Description:

Method	*Return values*	*Description*
saveSearch(String token, QueryParams params)	***Long***	*Saves search parameters.*
The variable queryName of the parameter params should have to be initialized.		

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");

            // Save the search to be used later
            qParams.setQueryName("sample search");
            okmSearch.saveSearch(null, qParams);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateSearch

Description:

Method	*Return values*	*Description*
updateSearch(String token, QueryParams params)	***void***	*Saves a search parameters.*

Only can be updated as a saved search created by the same user user who's executing the method.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (QueryParams qParams : okmSearch.getAllSearchs(null)) {
                if (qParams.getQueryName().equals("sample search")) {
                    // Change some value.
                    qParams.setName("admin*.html");
                    okmSearch.updateSearch(null, qParams);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getSearch

Description:

Method	*Return values*	*Description*
getSearch(String token, int qpId)	***QueryParams***	*Gets saved search.*

Only can be updated as a saved search created by the same user user who's executing the method.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            int qpId = 1; // Some valid search id
            QueryParams qParams = okmSearch.getSearch(null, qpId);
            System.out.println(qParams);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getAllSearchs

Description:

Method	*Return values*	*Description*
getAllSearchs(String token)	***List<QueryParams>***	*Retrieves a list of all saved search.*

Only can be updated as a saved search created by the same user who's executing the method.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (QueryParams qParams : okmSearch.getAllSearchs(null)) {
                System.out.println(qParams);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteSearch

Description:

Method	*Return values*	*Description*
deleteSearch(String token, long qpId)	***void***	*Deletes a saved search.*

Only can be updated as a saved search created by the same user user who's executing the method.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.util.ContextWrapper;

public class Test {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            int qpId = 1; // Some valid search id
            okmSearch.deleteSearch(null, qpId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getKeywordMap

Description:

Method	*Return values*	*Description*
getKeywordMap(String token, List<String> filter)	***Map<String, Integer>***	*Returns a map of keywords with its count value filtered by other keywords.*

Example:

- *Doc1.txt has keywords "test", "one", "two".*
- *Doc2.txt has keywords "test", "one"*
- *Doc3.txt has keywords "test", "three".*

The results filtering by "test" -> "one", "two", "three".

The results filtering by "one" -> "test", "two".

The results filtering by "two" -> "test", "one".

The results filtering by "three" -> "test".

The results filtering by "one" and "two" -> "test"

```
package com.openkm;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Map;

import com.openkm.api.OKMSearch;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            // All keywords without filtering
            System.out.println("Without filtering");
            Map<String, Integer> keywords = okmSearch.getKeywordMap(null, new ArrayLi
```

```
            for (String key : keywords.keySet()) {
                System.out.println(key + " is used :" + keywords.get(key));
            }

            // Keywords filtered
            System.out.println("Filtering");
            keywords = okmSearch.getKeywordMap(null, Arrays.asList("test"));

            for (String key : keywords.keySet()) {
                System.out.println(key + " is used :" + keywords.get(key));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getCategorizedDocuments

Description:

Method	*Return values*	*Description*
getCategorizedDocuments(String token, String categoryId)	***List<Document>***	*Retrieves a list of all documents related with a category.*
The values of the categoryId parameter should be a category folder UUID or path.		

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Document doc : okmSearch.getCategorizedDocuments(null, "5cbc35fc-23c
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getCategorizedFolders

Description:

Method	*Return values*	*Description*

getCategorizedFolders(String token, String categoryId)	***List<Folder>***	*Retrieves a list of all folders related to a category.*
The values of the categoryId parameter should be a category folder UUID or path.		

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Folder fld : okmSearch.getCategorizedFolders(null, "5cbc35fc-23c0-48
                System.out.println(fld);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getCategorizedMails

Description:

Method	***Return values***	***Description***
getCategorizedMails(String token, String categoryId)	***List<Mail>***	*Retrieves a list of all mails related to a category.*
The values of the categoryId parameter should be a category folder UUID or path.		

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Mail mail : okmSearch.getCategorizedMails(null, "5cbc35fc-23c0-48f5-
                System.out.println(mail);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

getCategorizedRecords

Description:

Method	*Return values*	*Description*
getCategorizedRecords(String token, String categoryId)	***List<Record>***	*Retrieves a list of all records related to a category.*
The values of the categoryId parameter should be a category folder UUID or path.		

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Record rec : okmSearch.getCategorizedRecords(null, "5cbc35fc-23c0-48
                System.out.println(rec);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getDocumentsByKeyword

Description:

Method	*Return values*	*Description*
getDocumentsByKeyword(String token, String keyword)	***List<Document>***	*Retrieves a list of all documents related to a keyword.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Document doc : okmSearch.getDocumentsByKeyword(null, "test")) {
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFoldersByKeyword

Description:

Method	*Return values*	*Description*
getFoldersByKeyword(String token, String keyword)	***List<Folder>***	*Retrieves a list of all folders related to a keyword.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Folder fld : okmSearch.getFoldersByKeyword(null, "test")) {
                System.out.println(fld);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMailsByKeyword

Description:

Method	*Return values*	*Description*
getMailsByKeyword(String token, String keyword)	***List<Mail>***	*Retrieves a list of all mails related to a keyword.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Mail;
```

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Mail mail : okmSearch.getMailsByKeyword(null, "test")) {
                System.out.println(mail);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRecordsByKeyword

Description:

Method	*Return values*	*Description*
getRecordsByKeyword(String token, String keyword)	***List<Record>***	*Retrieves a list of all records related to a keyword.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Record rec : okmSearch.getRecordsByKeyword(null, "test")) {
                System.out.println(rec);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getDocumentsByPropertyValue

Description:

Method	*Return values*	*Description*
getDocumentsByPropertyValue(String token, String group, String property, String value	***List<Document>***	*Retrieves a list of all documents related with a metadata field value.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Document doc : okmSearch.getDocumentsByPropertyValue(null, "okg:cons
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFoldersByPropertyValue

Description:

Method	*Return values*	*Description*
getFoldersByPropertyValue(String token, String group, String property, String value	***List<Folder>***	*Retrieves a list of all folders related with a metadata field value.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Folder;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Folder fld : okmSearch.getFoldersByPropertyValue(null, "okg:consulti
                System.out.println(fld);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMailsByPropertyValue

Description:

Method	*Return values*	*Description*

Method	Return values	Description
getMailsByPropertyValue(String token, String group, String property, String value	***List<Mail>***	*Retrieves a list of all mails related with a metadata field value.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Mail;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Mail mail : okmSearch.getMailsByPropertyValue(null, "okg:consulting"
                System.out.println(mail);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRecordsByPropertyValue

Description:

Method	Return values	Description
getRecordsByPropertyValue(String token, String group, String property, String value)	***List<Record>***	*Retrieves a list of all records related with a metadata field value.*

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.Record;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            for (Record rec : okmSearch.getRecordsByPropertyValue(null, "okg:consulti
                System.out.println(rec);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findMailPaginated

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
findMailPaginated(String token, Map<String, String> filter, String sortField, boolean sortReverse, int type, int offset, int limit)	***ResultSet***	*Returns a list of paginated mail results filtered by the values of the parameters.*

Allowed filter key values are:

- *Mail.SENT_DATE*
- *Mail.SUBJECT*
- *Mail.TO*

Send data value format must be:

```
ISO8601.formatBasic(from)+","+ISO8601.formatBasic(to)
```

Allowed sortField values are:

- *"okm:" for a non sorted.*
- *Mail.TYPE*
- *Mail.SUBJECT*
- *Mail.FROM*
- *Mail.SENT_DATE*
- *Mail.SIZE*
- *Mail.REPLY*
- *Mail.TO*
- *Mail.CC*
- *Mail.BCC*
- *Mail.CONTENT*

Allowed type values are:

- *OKMSearch.TYPE_MAIL_ALL*
- *OKMSearch.TYPE_MAIL_WITHOUT_ATTACHMENT*
- *OKMSearch.TYPE_MAIL_WITH_ATTACHMENT*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

```
package com.openkm;

import java.util.HashMap;
import java.util.Map;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            Map<String, String> filter = new HashMap<>();
            String sortField = "okm:";
            int type = OKMSearch.TYPE_MAIL_ALL;
            ResultSet rs = okmSearch.findMailPaginated(null, filter, sortField, false
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findPaginated

Description:

Method	*Return values*	*Description*

findPaginated(String token, QueryParams params, String sortField, boolean sortReverse, int offset, int limit)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");
            String sortField = "okm:";
            boolean sortReverse = false;
            ResultSet rs = okmSearch.findPaginated(null, qParams, sortField, sortReve
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        }
    }
```

findPaginated

Description:

Method	*Return values*	*Description*
findPaginated(String token, QueryParams params, String sortField, boolean sortReverse, int offset, int limit, String propertiesPlugin)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams parameter.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");
            String sortField = "okm:";
            boolean sortReverse = false;
            ResultSet rs = okmSearch.findPaginated(null, qParams, sortField, sortReve
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findWithMetadata

Description:

Method	*Return values*	*Description*
findWithMetadata(String token, QueryParams params, String propertiesPlugin, List<String> groups)	***List<QueryResult>***	*Returns a list of results filtered by the values of the queryParams parameter.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

The parameter "groups" is the name metadata group.

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only a few Objects variables. If it's your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.db.bean.QueryParams;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");

            List<String> groups = new ArrayList<>();
            groups.add("okg:tpl");

            for (QueryResult qr : okmSearch.findWithMetadata(null, qParams, "com.open
                    groups)) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findWithMetadataPaginated

Description:

Method	*Return values*	*Description*
findWithMetadataPaginated(String token, QueryParams params, int offset, int limit, String propertiesPlugin, List<String> groups)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" used to skip that many results before the beginning to return results.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

The parameter "groups" is the name metadata group.

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");

            List<String> groups = new ArrayList<>();
            groups.add("okg:tpl");

            ResultSet rs = okmSearch.findWithMetadataPaginated(null, qParams, 0, 10,
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findWithMetadataPaginated

Description:

Method	*Return values*	*Description*
findWithMetadataPaginated(String token, QueryParams params, String sortField, boolean sortReverse, int offset, int limit, String propertiesPlugin, List<String> groups)	***ResultSet***	*Returns a list of paginated results filtered by the values of the queryParams*

		parameter.

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" used to skip that many results before the beginning to return results.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

The parameter "groups" is the name metadata group.

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.db.bean.QueryParams;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            QueryParams qParams = new QueryParams();
            qParams.setDomain(QueryParams.DOCUMENT + QueryParams.FOLDER);
            qParams.setName("test*");
            String sortField = "okm:";
            boolean sortReverse = false;

            List<String> groups = new ArrayList<>();
```

```
            groups.add("okg:tpl");

            ResultSet rs = okmSearch.findWithMetadataPaginated(null, qParams, sortFie
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQueryPaginated

Description:

Method	*Return values*	*Description*
findByQueryPaginated(String token, String query, String sortField, boolean sortReverse, int offset, int limit)	***ResultSet***	*Returns a list of paginated results filtered by the value of the query parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
```

```
import com.openkm.bean.ResultSet;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            String sortField = "okm:";
            boolean sortReverse = false;

            ResultSet rs = okmSearch.findByQueryPaginated(null, "text:grial AND name:
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findByQueryPaginated

Description:

Method	*Return values*	*Description*
findByQueryPaginated(String token, String query, String sortField, boolean sortReverse, int offset, int limit, String propertiesPlugin)	***ResultSet***	*Returns a list of paginated results filtered by the value of the query parameter.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

The parameter "propertiesPlugin" must be a canonical class name of the class which implements the NodeProperties interface.

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*

- *offset=10*

Retrieving entire Objects (Document, Folder, Record, Mail) from REST can take a lot of time - marshalling and unmarshalling process - while you are only interesting on using only few Objects variables. If its your case you can use NodeProperties classes for retrieving the Object variables what you really need.

Example:

```
package com.openkm;

import com.openkm.api.OKMSearch;
import com.openkm.bean.QueryResult;
import com.openkm.bean.ResultSet;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMSearch okmSearch = ContextWrapper.getContext().getBean(OKMSearch.class
            String sortField = "okm:document";
            boolean sortReverse = false;

            ResultSet rs = okmSearch.findByQueryPaginated(null, "text:grial AND name:
                    "com.openkm.plugin.properties.SimpleNodeProperties");
            System.out.println("Total results:" + rs.getTotal());

            for (QueryResult qr : rs.getResults()) {
                System.out.println(qr);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMStamp

Methods

calculateStampCoordinates

Description:

Method	*Return values*	*Description*
calculateStampCoordinates(String token, long imgToStampWidth, long imgToStampHeight, long floatingDivWidth,long floatingDivHeight, String exprX, String exprY, String stampType, String stampAlign)	***StampCoordinates***	*Return the calculated stamp coordinates.*

In Expr. X and Expr. Y input fields you can put more than a simple number. Currently, the following macros are defined:

- *IMAGE_WIDTH*
- *IMAGE_HEIGHT*
- *PAGE_WIDTH*
- *PAGE_HEIGHT*
- *PAGE_CENTER*
- *PAGE_MIDDLE*

So to center a stamp in the page you can use:

Expr. X	`PAGE_CENTER`
Expr. Y	`PAGE_MIDDLE`

*The parameter **stampAlign.***

Available values:

- *text*
- *image*

*The parameter **stampAlign** is the Text Align*

Available values:

- *left*
- *center*
- *right*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.StampCoordinates;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            // Parameters
            long imgToStampWidth = 100;
            long imgToStampHeight = 100;
            long floatingDivWidth = 250;
            long floatingDivHeight = 300;
            String exprX = "PAGE_CENTER - IMAGE_WIDTH / 2";
            String exprY = "PAGE_MIDDLE - IMAGE_HEIGHT / 2";
            String stampType = "text";
            String stampAlign = "center";
            StampCoordinates stampCoordinates = okmStamp.calculateStampCoordinates(nu
                    floatingDivWidth, floatingDivHeight, exprX, exprY, stampType, sta
            System.out.println("X = " + stampCoordinates.getX() + ", Y = " + stampCoo
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

calculateStampExpressions

Description:

Method	*Return values*	*Description*
calculateStampExpressions(String token, long imgToStampWidth, long imgToStampHeight, long posX, long posY)	***StampExpressions***	*Return the calculated stamp expressions for X and Y.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.StampExpressions;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            // Parameters
            long imgToStampWidth = 100;
            long imgToStampHeight = 100;
            long posX = 10;
            long posY = 50;
            StampExpressions stampExpressions = okmStamp.calculateStampExpressions(nu
            System.out.println("X = " + stampExpressions.getExprX() + ", Y = " + stam
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

stampDocument

Description:

Method	*Return values*	*Description*
stampDocument(String token, long id, int type, String uuid)	***void***	*Stamp the document*

The parameter ***type.***

- *Value 0 for type text*
- *Value 1 for type image*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            long stampId = 1;
            okmStamp.stampDocument(null, stampId, 1, "babe24df-c443-49a5-9453-39dfc9b
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

stampImageCustom

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
void stampImageCustom(String token, String uuid, String id, String exprX, String exprY, String range, String personalStampUuid)	***void***	*Stamp a document with a custom image.*

> *The **personalStampUuid** must be a valid document **UUID**. The document must be an image (jpg or png).*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            long stampId = 3;
            String exprX = "PAGE_CENTER - IMAGE_WIDTH / 2";
            String exprY = "PAGE_MIDDLE - IMAGE_HEIGHT / 2";
            okmStamp.stampImageCustom(null, "babe24df-c443-49a5-9453-39dfc9b27924", s
                    "928de61e-6ceb-4f94-bf20-9ba2405c5cee");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

stampImageCustom

Description:

Method	*Return values*	*Description*
void stampImageCustom(String token, String uuid, String id, String exprX, String exprY, String range, byte[] image)	***void***	*Stamp a document with a custom image.*

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileInputStream;
```

```
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            long stampId = 3;
            String exprX = "PAGE_CENTER - IMAGE_WIDTH / 2";
            String exprY = "PAGE_MIDDLE - IMAGE_HEIGHT / 2";
            File file = new File("/hombe/openkm/logo.png");
            InputStream in = new FileInputStream(file);
            byte[] image = IOUtils.toByteArray(in);
            IOUtils.closeQuietly(in);
            okmStamp.stampImageCustom(null, "babe24df-c443-49a5-9453-39dfc9b27924", s
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAllStamps

Description:

Method	*Return values*	*Description*
getAllStamps(String token)	***List<StampItem>***	*Returns a list of stamps*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.StampItem;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            List<StampItem> result = okmStamp.getAllStamps();
            for (StampItem stampItem : result) {
                System.out.println(stampItem.getName());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

getStampTextByPk

Description:

Method	*Return values*	*Description*
getStampTextByPk(String token, long itemId, String docId)	***StampText***	*Return the stamp Object by Id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.extension.db.bean.StampText;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            int stampId = 1;
            StampText stampText = okmStamp.getStampTextByPk(null, stampId, "928de61e-
            System.out.println(stampText.getName());
            System.out.println(stampText.getDescription());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getStampImageByPk

Description:

Method	*Return values*	*Description*
getStampImageByPk(String token, long id, String docId)	***StampImage***	*Return the stamp Object by Id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.extension.db.bean.StampImage;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            int stampId = 3;
            StampImage stampImage = okmStamp.getStampImageByPk(null, stampId, "928de6
```

```
            System.out.println(stampImage.getName());
            System.out.println(stampImage.getDescription());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

calculateFlyImageDimensions

Description:

Method	*Return values*	*Description*
calculateFlyImageDimensions(String token, String uuid, long imgToStampWidth, long imgToStampHeight,long floatingImageWidth, long floatingImageHeight, int pageNumber)	***StampImageSize***	*Return the calculated image dimensions of the stamp.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.StampImageSize;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            // Parameters
            long imgToStampWidth = 100;
            long imgToStampHeight = 100;
            long floatingImageWidth = 300;
            long floatingImageHeight = 300;
            int pageNumber = 2;

            StampImageSize stampImageSize = okmStamp.calculateFlyImageDimensions(null
            System.out.println(stampImageSize);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPersonalStamps

Description:

Method	Return values	Description
getPersonalStamps(String token)	**List<Document>**	Returns a list of personal seals.

> Return a list of documents of type image (jpg or png) from /okm:templates/ userId/stamps

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.api.OKMStamp;
import com.openkm.bean.Document;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            List<Document> result = okmStamp.getPersonalStamps(null);
            for (Document doc : result) {
                System.out.println(doc);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getPersonalStampImage

Description:

Method	Return values	Description
getPersonalStampImage(String token, String uuid, long id)	**StampPersonalImage**	Return the personal seal processed.

> When stamping a document with an image, the image also can have a text layer.
>
> This method processes the image and returns an image with the text layer.
>
> The **uuid** must be one of the values returned by the method getPersonalStamps.

Example:

```
package com.openkm;

import com.openkm.api.OKMStamp;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.StampPersonalImage;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStamp okmStamp = ContextWrapper.getContext().getBean(OKMStamp.class);
            long stampId = 1;
            StampPersonalImage stampPersonalImage = okmStamp.getPersonalStampImage(nu
            System.out.println(stampPersonalImage);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMStats

Basics

*On all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

getDocumentsByContext

Description:

Method	*Return values*	*Description*
getDocumentsByContext(String token)	***StatsInfo***	*Return stats information about documents by context.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStats;
import com.openkm.bean.StatsInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStats okmStats = ContextWrapper.getContext().getBean(OKMStats.class);
            StatsInfo sInfo = okmStats.getDocumentsByContext(null);
            System.out.println(sInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getDocumentsSizeByContext

Description:

Method	*Return values*	*Description*

getDocumentsByContext(String token)	***StatsInfo***	*Return stats information about documents size by context.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStats;
import com.openkm.bean.StatsInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStats okmStats = ContextWrapper.getContext().getBean(OKMStats.class);
            StatsInfo sInfo = okmStats.getDocumentsSizeByContext(null);
            System.out.println(sInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFoldersByContext

Description:

Method	*Return values*	*Description*
getFoldersByContext(String token)	***StatsInfo***	*Return stats information about folders by context.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStats;
import com.openkm.bean.StatsInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStats okmStats = ContextWrapper.getContext().getBean(OKMStats.class);
            StatsInfo sInfo = okmStats.getFoldersByContext(null);
            System.out.println(sInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getMailsByContext

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	Return values	Description
getMailsByContext(String token)	***StatsInfo***	*Return stats information about mails by context.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStats;
import com.openkm.bean.StatsInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStats okmStats = ContextWrapper.getContext().getBean(OKMStats.class);
            StatsInfo sInfo = okmStats.getMailsByContext(null);
            System.out.println(sInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getRecordsByContext

Description:

Method	Return values	Description
getRecordsByContext(String token)	***StatsInfo***	*Return stats information about records by context.*

Example:

```
package com.openkm;

import com.openkm.api.OKMStats;
import com.openkm.bean.StatsInfo;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMStats okmStats = ContextWrapper.getContext().getBean(OKMStats.class);
            StatsInfo sInfo = okmStats.getRecordsByContext(null);
            System.out.println(sInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMTask

Basics

Task fields description:

Field	*Type*	*Description*	*Mandatory*
Id	***Long***	*Internal task id.* *It is automatically set by the application.*	*Not applicable.*
Owner	***String***	*It contains the OpenKM userId.* *It is automatically set by the application.*	*Not applicable.*
Subject	***String***	*The topic of the task.*	*Yes.*
Description	***String***	*The description of the task.*	*No.*
Start	***Calendar***	*When the task might start.*	*Yes.*
End	***Calendar***	*When the task might ending.*	*No.*
Status	***TaskStatus***	*The task status.* *You administering the list of status values.*	*Yes.*
Project	***TaskProject***	*The project related to the task.* *You administering the list of project values.*	*Yes.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Type	***TaskType***	*The task type.* *You administering the list of task type values.*	*Yes.*
Progress	***Integer***	*The numeric progress status.* *The range of allowed values are from 0 to 100. Where 100% indicates a completed task.*	*Yes.*
RepeatGroup	***Long***	*When task is repeated along time is a member of a group. The group is identified by an unique repeat group id.*	*No.*
ReminderStartValue	***Integer***	*How many days before the task starting, the system might send a mail notification to the user.*	*No.*
ReminderEndValue	***Integer***	*How many days before the task ending, the system might send a mail notification to the user.*	*No.*
ReminderStartUnit	***String***	*Reminder start units.* *Allowed values:* • *"m" for minutes.* • *"h" for hours.* • *"d" for days.*	*Mandatory when **ReminderStartValue** greater than 0.*
ReminderEndUnit	***String***	*Reminder end units.* *Allowed values:* • *"m" for minutes.* • *"h" for hours.* • *"d" for days.*	*Mandatory when **ReminderEndValue** greater than 0.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Users	***Set<String>***	*Collection of assigned users to the task.*	*No. But At least should be a user or role assigned.*
Roles	***Set<String>***	*Collection of assigned roles to the task.*	*No. But At least should be a user or role assigned.*
Documents	***Set<String>***	*List of UUID's of the related documents.*	*No.*
Folders	***Set<String>***	*List of UUID's of the related folders.*	*No.*
Mails	***Set<String>***	*List of UUID's of the related mails.*	*No.*
Records	***Set<String>***	*List of UUID's of the related records.*	*No.*

*On all methods, there's a parameter named "**token**". When accessing application across REST, the login process returns a token. The token is used to identify the user on all the exposed methods. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

create

Description:

Method	*Return values*	*Description*
create(String token, Task task)	***Task***	*Create a new task.*

Example:

```
package com.openkm;

import java.util.Calendar;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.bean.TaskProject;
import com.openkm.bean.TaskStatus;
import com.openkm.bean.TaskType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            Task task = new Task();
            task.setSubject("task subject");
            task.setDescription("description");
            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, 15);
            task.setStart(start);
            long taskStatusId = 1; // A valid task status id
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            TaskStatus status = okmTask.findStatus(null, taskStatusId);
            TaskProject project = okmTask.findProject(null, projectId);
            TaskType type = okmTask.findType(null, typeId);
            task.setStatus(status);
            task.setProject(project);
            task.setType(type);
            task.setProgress(0);
            okmTask.create(null, task);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

update

Description:

Method	*Return values*	*Description*
update(String token, Task task)	***Task***	*Update a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long taskId = 1; // A valid task id
            Task task = okmTask.find(null, taskId);
```

```
            task.setSubject("subject modified");
            okmTask.update(null, task);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

find

Description:

Method	*Return values*	*Description*
find(String token, Task task)	***Task***	*Find a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long taskId = 1; // A valid task id
            Task task = okmTask.find(null, taskId);
            System.out.println(task);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

delete

Description:

Method	*Return values*	*Description*
delete(String token, Task task)	***void***	*Delete a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long taskId = 1; // A valid task id
            okmTask.delete(null, taskId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getByProject

Description:

Method	*Return values*	*Description*
getByProject(String token, long projectId)	***List<Task>***	*Retrieve a list of task related with a project.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            for (Task task : okmTask.getByProject(null, projectId)) {
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getByType

Description:

Method	*Return values*	*Description*
getByType(String token, long typeId)	***List<Task>***	*Retrieve a list of task related with a task type.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long typeId = 1; // A valid type id
            for (Task task : okmTask.getByType(null, typeId)) {
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAssigned

Description:

Method	*Return values*	*Description*
getAssigned(String token, long projectId, long typeId, long statusId, String orderColumn)	***List<Task>***	*Retrieve a list of tasks assigned to a user and filtered.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "subject"; // A valid TaskManagerTask class paramete
            for (Task task : okmTask.getAssigned(null, projectId, typeId, statusId, o
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAssigned

Description:

Method	*Return values*	*Description*
getAssigned(String token, long projectId, long typeId, long statusId, String orderColumn, boolean orderAsc, int offset, int limit)	***List<Task>***	*Retrieve a list of tasks assigned to a user, filtered and paginated.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*
- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*

- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "start"; // A valid TaskManagerTask class parameter
            for (Task task : okmTask.getAssigned(null, projectId, typeId, statusId, o
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countAssigned

Description:

Method	*Return values*	*Description*
countAssigned(String token, long projectId, long typeId, long statusId)	***Long***	*Return the number of tasks assigned to a user.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            long total = okmTask.countAssigned(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getActive

Description:

Method	*Return values*	*Description*
getActive(String token, long projectId, long typeId, long statusId, String orderColumn)	***List<Task>***	*Retrieve a list of active tasks assigned to a user and filtered.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
```

```
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "start"; // A valid TaskManagerTask class parameter
            for (Task task : okmTask.getActive(null, projectId, typeId, statusId, ord
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getActive

Description:

Method	*Return values*	*Description*
getActive(String token, long projectId, long typeId, long statusId, String orderColumn, boolean orderAsc, int offset, int limit)	***List<Task>***	*Retrieve a list of active tasks assigned to a user, filtered and paginated.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*
- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "start"; // A valid TaskManagerTask class parameter
            for (Task task : okmTask.getActive(null, projectId, typeId, statusId, ord
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countActive

Description:

Method	*Return values*	*Description*
countActive(String token, long projectId, long typeId, long statusId)	***Long***	*Return the number of active tasks of a user.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*

- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            long total = okmTask.countAssigned(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFinished

Description:

Method	*Return values*	*Description*
getFinished(String token, long projectId, long typeId, long statusId, String orderColumn)	***List<Task>***	*Retrieve a list of finished tasks assigned to a user and filtered.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "start"; // A valid TaskManagerTask class parameter
            for (Task task : okmTask.getFinished(null, projectId, typeId, statusId, o
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFinished

Description:

Method	*Return values*	*Description*
getFinished(String token, long projectId, long typeId, long statusId, String orderColumn, boolean orderAsc, int offset, int limit)	***List<Task>***	*Retrieve a list of finished tasks assigned to a user, filtered and paginated.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*
- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "start"; // A valid TaskManagerTask class parameter
            for (Task task : okmTask.getFinished(null, projectId, typeId, statusId, o
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countFinished

Description:

Method	*Return values*	*Description*
countFinished(String token, long projectId, long typeId, long statusId)	***Long***	*Return the number of finished tasks of a user.*

Filter parameters description:

- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            long total = okmTask.countFinished(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findStatus

Description:

Method	*Return values*	*Description*
findStatus(String token, long statusId)	***TaskStatus***	*Return a task status object by id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskStatus;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long statusId = 1; // A valid status id
            TaskStatus ts = okmTask.findStatus(null, statusId);
```

```
            System.out.println(ts);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findProject

Description:

Method	*Return values*	*Description*
findProject(String token, long projectId)	***TaskProject***	*Return a task project object by id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskProject;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            TaskProject tp = okmTask.findProject(null, projectId);
            System.out.println(tp);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findType

Description:

Method	*Return values*	*Description*
findType(String token, long typeId)	***TaskType***	*Return a task type object by id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskType;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long typeId = 1; // A valid type id
            TaskType tt = okmTask.findType(null, typeId);
            System.out.println(tt);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findNote

Description:

Method	*Return values*	*Description*
findNote(String token, long noteId)	***TaskNote***	*Return a task note object by id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskNote;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long noteId = 2; // A valid note id
            TaskNote tn = okmTask.findNote(null, noteId);
            System.out.println(tn);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getStatuses

Description:

Method	*Return values*	*Description*
getStatuses(String token)	***List<TaskStatus>***	*Retrieve a list of all the task status.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskStatus;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            for (TaskStatus ts : okmTask.getStatuses(null)) {
                System.out.println(ts);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProjects

Description:

Method	*Return values*	*Description*
getProjects(String token, boolean filterByActive)	***List<TaskProject>***	*Retrieve a list of all the task projects.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskProject;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            for (TaskProject tp : okmTask.getProjects(null, true)) {
                System.out.println(tp);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getTypes

Description:

Method	*Return values*	*Description*

getTypes(String token, boolean filterByActive)	***List<TaskType>***	*Retrieve a list of all the task types.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            for (TaskType tt : okmTask.getTypes(null, true)) {
                System.out.println(tt);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNotes

Description:

Method	*Return values*	*Description*
getNotes(String token, long taskId)	***List<TaskNote>***	*Retrieve a list of all the notes of a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskNote;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long taskId = 1; // A valid task id
            for (TaskNote tn : okmTask.getNotes(null, taskId)) {
                System.out.println(tn);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

createStatus

Description:

Method	*Return values*	*Description*
createStatus(String token, String name, boolean finish)	***TaskStatus***	*Create a new task status.*

> *Depending on your logic, several status can be ending status. For example, status named "closed" or "cancelled". The boolean parameter "finish" indicates if your status in an "ending status".*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskStatus;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            TaskStatus ts = okmTask.createStatus(null, "cancelled", true);
            System.out.println(ts);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createProject

Description:

Method	*Return values*	*Description*
createProject(String token, String name, String description, boolean active)	***TaskProject***	*Create a new task project.*

> *The boolean parameter "active", indicates if your project is active or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskProject;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            TaskProject tp = okmTask.createProject(null, "Project one", "Description"
            System.out.println(tp);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createType

Description:

Method	*Return values*	*Description*
createType(String token, String name, String description, boolean active)	***TaskType***	*Create a new task type.*

> *The boolean parameter "active", indicates if your type is active or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            TaskType tt = okmTask.createType(null, "Type one", "Description", true);
            System.out.println(tt);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createNote

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
createNote(String token, long taskId, String text)	**TaskNote**	*Create a new note to a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskNote;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long taskId = 1; // A valid task id
            TaskNote tn = okmTask.createNote(null, taskId, "New note");
            System.out.println(tn);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateStatus

Description:

Method	Return values	Description
updateStatus(String token, long id, String name, boolean finish)	**TaskStatus**	*Update a task status.*

Depending on your logic, several status can be ending status. For example, status named "closed" or "cancelled". The boolean parameter "finish" indicates if your status in an "ending status".

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskStatus;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long statusId = 1; // A valid task id
```

```
            TaskStatus ts = okmTask.findStatus(null, statusId);
            System.out.println(ts);
            ts = okmTask.updateStatus(null, statusId, "cancelled", true);
            System.out.println(ts);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateProject

Description:

Method	*Return values*	*Description*
updateProject(String token, long id, String name, String description, boolean active)	***TaskProject***	*Update a task project.*

The boolean parameter "active", indicates if your project is active or not.

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskProject;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 4; // A valid project id
            TaskProject tp = okmTask.findProject(null, projectId);
            System.out.println(tp);
            tp = okmTask.updateProject(null, projectId, "cancelled", "Description", f
            System.out.println(tp);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateType

Description:

Method	*Return values*	*Description*

updateType(String token, long id, String name, String description, boolean active)	***TaskType***	*Update a task type.*

> *The boolean parameter "active", indicates if your type is active or not.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskType;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long typeId = 4; // A valid type id
            TaskType tp = okmTask.findType(null, typeId);
            System.out.println(tp);
            tp = okmTask.updateType(null, typeId, "Type one", "Description", false);
            System.out.println(tp);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

updateNote

Description:

Method	*Return values*	*Description*
updateNote(String token, long id, String text)	***TaskNote***	*Update note of a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.TaskNote;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long noteId = 1; // A valid note id
```

```
            TaskNote tn = okmTask.updateNote(null, noteId, "Note updated");
            System.out.println(tn);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteStatus

Description:

Method	*Return values*	*Description*
deleteStatus(String token, long statusId)	***void***	*Delete a task status.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long statusId = 4; // A valid status id
            okmTask.deleteStatus(null, statusId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteProject

Description:

Method	*Return values*	*Description*
deleteProject(String token, long projectId)	***void***	*Delete a task project.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 5; // A valid project id
            okmTask.deleteProject(null, projectId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteType

Description:

Method	*Return values*	*Description*
deleteType(String token, long typeId)	***void***	*Delete a task type.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long typeId = 4; // A valid type id
            okmTask.deleteType(null, typeId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteNote

Description:

Method	*Return values*	*Description*
deleteNote(String token, long noteId)	***void***	*Delete note of a task.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;
```

```
public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long noteId = 1; // A valid note id
            okmTask.deleteNote(null, noteId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getByNode

Description:

Method	*Return values*	*Description*
getByNode(String token, String nodeUuid)	***List<Task>***	*Return the list of tasks assigned to a specific node.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String uuid = "d168924d-e801-426d-a222-00124a1461bd"; // A valid uuid
            for (Task task : okmTask.getByNode(null, uuid)) {
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAssigned

Description:

Method	*Return values*	*Description*
getAssigned(String token, String subject, int progress, long projectId, long typeId, long statusId,	***TaskList***	*Retrieve a list of tasks assigned to*

Calendar lastModified, Calendar start, Calendar end, String orderColumn, boolean orderAsc, int offset, int limit)		*a user, filtered and paginated.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*
- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.TaskList;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "subject"; // A valid TaskManagerTask class paramete
            boolean orderAsc = true;
            int offset = 0;
            int limit = 10;
            Calendar lastModified = Calendar.getInstance();
            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);
            Calendar end = Calendar.getInstance();

            TaskList taskList = okmTask.getAssigned(null, subject, 10, projectId, typ
                    orderColumn, orderAsc, offset, limit);
            for (Task task : taskList.getTasks()) {
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countAssigned

Description:

Method	*Return values*	*Description*
countAssigned(String token, String subject, int progress, long projectId, long typeId, long statusId, Calendar lastModified, Calendar start, Calendar end)	***long***	*Return the number of tasks assigned to a user.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The progress parameter is the progress of the task.*
- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;
```

```
import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            int progress = 10;
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            Calendar lastModified = Calendar.getInstance();

            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);

            Calendar end = Calendar.getInstance();

            long total = okmTask.countAssigned(null, subject, progress, projectId, ty
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getActive

Description:

Method	*Return values*	*Description*
getActive(String token, String subject, int progress, long projectId, long typeId, long statusId, Calendar lastModified, Calendar start, Calendar end, String orderColumn, boolean orderAsc, int offset, int limit)	***TaskList***	*Retrieve a list of active tasks assigned to a user, filtered and paginated.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The progress parameter is the progress of the task.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc*

documentation.

- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

For example if your query has 1000 results, but you only want to return the first 10, you should use these values:

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.TaskList;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "progress"; // A valid TaskManagerTask class paramet
            boolean orderAsc = true;
            int offset = 0;
            int limit = 10;
            Calendar lastModified = Calendar.getInstance();
            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);
            Calendar end = Calendar.getInstance();

            TaskList taskList = okmTask.getActive(null, subject, 10, projectId, typeI
                    orderColumn, orderAsc, offset, limit);
            for (Task task : taskList.getTasks()) {
```

```
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countActive

Description:

Method	*Return values*	*Description*
countActive(String token, String subject, int progress, long projectId, long typeId, long statusId, Calendar lastModified, Calendar start, Calendar end)	***long***	*Return the number of active tasks to a user.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The progress parameter is progress of the task.*
- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            int progress = 10;
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            Calendar lastModified = Calendar.getInstance();

            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);
```

```
            Calendar end = Calendar.getInstance();

            long total = okmTask.countActive(null, subject, progress, projectId, type
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFinished

Description:

Method	*Return values*	*Description*
getFinished(String token, String subject, int progress, long projectId, long typeId, long statusId, Calendar lastModified, Calendar start, Calendar end, String orderColumn, boolean orderAsc, int offset, int limit)	***TaskList***	*Retrieve a list of finished tasks assigned to a user, filtered and paginated.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The progress parameter is the progress of the task.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*
- *The statusId parameters must be a valid status id.*
- *The orderColumn parameter must be a valid parameters of the TaskManagerTask class. Common used parameters are "subject", "start", "end", "progress", "owner". More information at javadoc documentation.*
- *The orderAsc parameter orders ascending or descending.*

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" says to skip that many results before the beginning to return results.*

For example, if your query has 1000 results, but you only want to return the first 10, you should use these values:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *limit=10*
- *offset=0*

Now suppose you want to show the results from 11-20, you should use these values:

- *limit=10*
- *offset=10*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.bean.Task;
import com.openkm.util.ContextWrapper;
import com.openkm.ws.rest.util.TaskList;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            String orderColumn = "end"; // A valid TaskManagerTask class parameter na
            boolean orderAsc = true;
            int offset = 0;
            int limit = 10;
            Calendar lastModified = Calendar.getInstance();
            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);
            Calendar end = Calendar.getInstance();

            TaskList taskList = okmTask.getFinished(null, subject, 10, projectId, typ
                    orderColumn, orderAsc, offset, limit);
            for (Task task : taskList.getTasks()) {
                System.out.println(task);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

countFinished

Description:

Method	*Return values*	*Description*

countFinished(String token, String subject, int progress, long projectId, long typeId, long statusId, Calendar lastModified, Calendar start, Calendar end)	***long***	*Return the number of finished tasks of a user.*

Filter parameters description:

- *The subject parameter the topic of the task.*
- *The progress parameter is the progress of the task.*
- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            String subject = "test subject";
            int progress = 10;
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id
            Calendar lastModified = Calendar.getInstance();

            Calendar start = Calendar.getInstance();
            start.add(Calendar.DATE, -30);

            Calendar end = Calendar.getInstance();

            long total = okmTask.countFinished(null, subject, progress, projectId, ty
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getAssignedCount

Description:

Method	*Return*	*Description*

		values
getAssignedCount(String token, long statusId, long projectId, long typeId)	***long***	*Return the number of tasks assigned to a user.*

Filter parameters description:

- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id

            long total = okmTask.getAssignedCount(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getActiveCount

Description:

Method	*Return values*	*Description*
getActiveCount(String token, long statusId, long projectId, long typeId)	***Long***	*Return the number of active tasks of a user.*

Filter parameters description:

- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id

            long total = okmTask.getActiveCount(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFinishedCount

Description:

Method	*Return values*	*Description*
getFinishedCount(String token, long statusId, long projectId, long typeId)	***Long***	*Return the number of finished tasks of a user.*

Filter parameters description:

- *The statusId parameters must be a valid status id.*
- *The projectId parameter must be a valid project id.*
- *The typeId parameters must be a valid type id.*

Example:

```
package com.openkm;

import com.openkm.api.OKMTask;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMTask okmTask = ContextWrapper.getContext().getBean(OKMTask.class);
            long projectId = 1; // A valid project id
            long typeId = 1; // A valid type id
            long statusId = 1; // A valid status id

            long total = okmTask.getFinishedCount(null, projectId, typeId, statusId);
            System.out.println(total);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMUserConfig

*On all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

setHome

Description:

Method	Return values	Description
setHome(String token, String nodeId)	**void**	*Sets the user home.*

By default user home value is "/okm:root".

Example:

```
package com.openkm;

import com.openkm.api.OKMUserConfig;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMUserConfig okmUserConfig = ContextWrapper.getContext().getBean(OKMUser
            okmUserConfig.setHome(null, "/okm:root/test");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getConfig

Description:

Method	*Return values*	*Description*
getConfig(String token)	***UserConfig***	*Return the user configuration.*

Returned values are:

- *The userId.*
- *The home path.*
- *The home type (folder, document, mail or record).*
- *The home node (the UUID).*

Example:

```
package com.openkm;

import com.openkm.api.OKMUserConfig;
import com.openkm.db.bean.UserConfig;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            OKMUserConfig okmUserConfig = ContextWrapper.getContext().getBean(OKMUser
            UserConfig uc = okmUserConfig.getConfig(null);
            System.out.println(uc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

OKMWorkflow

*On all methods you'll see parameter named "**token**". Because the Cron Task are executed in background without authentication, the methods used in this scenario might use the token parameter. From default application execution context you must use "**null**" value what indicates to the application must use the "**user session**".*

*On special cases you might be "**promoted as Administrator**" using the "**administrator token**".*

```
String systemToken = DbSessionManager.getInstance().getSystemToken();
```

Methods

registerProcessDefinition

Description:

Method	*Return values*	*Description*
registerProcessDefinition(String token, InputStream is)	***void***	*Registers a new workflow.*

Example:

```
package com.openkm;

import java.io.FileInputStream;
import java.io.InputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            InputStream is = new FileInputStream("/opt/files/Purchase.par");
            OKMWorkflow.getInstance().registerProcessDefinition(null, is);
            IOUtils.closeQuietly(is);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteProcessDefinition

Description:

Method	*Return values*	*Description*

deleteProcessDefinition(String token, long processDefinitionId)	***void***	*Deletes a workflow.*
The parameter processDefinitionId value is a valid workflow process definition.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long pdId = 5; // Valid workflow process definition
            OKMWorkflow.getInstance().deleteProcessDefinition(null, pdId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProcessDefinitionImage

Description:

Method	*Return values*	*Description*
getProcessDefinitionImage(String token, long processDefinitionId, String node)	***byte[]***	*Returns a workflow image process definition.*
The parameter processDefinitionId value is a valid workflow process definition. *The parameter node is a node with a workflow in execution.*		

Example:

```
package com.openkm;

import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileOutputStream;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long pdId = 5; // Valid workflow process definition
```

```
            byte[] img = OKMWorkflow.getInstance().getProcessDefinitionImage(null, pd
            ByteArrayInputStream in = new ByteArrayInputStream(img);
            FileOutputStream out = new FileOutputStream(new File("/home/test/img.jpg"
            IOUtils.copy(in, out);
            IOUtils.closeQuietly(in);
            IOUtils.closeQuietly(out);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProcessDefinitionForms

Description:

Method	*Return values*	*Description*
Map<String, List<FormElement>> getProcessDefinitionForms(String token, long processDefinitionId)	***Map<String, List<FormElement>>***	*Returns a map with all workflow form definitions.*
The parameter processDefinitionId value is a valid workflow process definition.		

Example:

```
package com.openkm;

import java.util.List;
import java.util.Map;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.form.FormElement;

public class Test {
    public static void main(String[] args) {
        try {
            long pdId = 5; // Valid workflow process definition
            Map<String,List<FormElement>> map = OKMWorkflow.getInstance().getProcessD
            for (String key : map.keySet()) {
                System.out.println("group:"+key);
                for (FormElement fe : map.get(key)) {
                    System.out.println(fe);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

runProcessDefinition

Description:

Method	*Return values*	*Description*
runProcessDefinition(String token, long processDefinitionId, String uuid, List<FormElement> variables)	***ProcessInstance***	*Executes a workflow on a node.*
The parameter processDefinitionId value is a valid workflow process definition. *The parameter uuid can be any document, mail, folder or record UUID.* *The parameter values are form element values needed for starting the workflow (not all workflows need form values for starting).*		

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.form.FormElement;
import com.openkm.bean.form.Input;
import com.openkm.bean.form.TextArea;

public class Test {
    public static void main(String[] args) {
        try {
            long pdId = 8041; // Some valid workflow process definition id
            List<FormElement> feList = new ArrayList<>(); // Case as part of starting
            Input price = new Input();
            price.setName("price");
            price.setValue("1000");
            feList.add(price);
            TextArea textArea = new TextArea();
            textArea.setName("description");
            textArea.setValue("some description here");
            feList.add(textArea);
            OKMWorkflow.getInstance().runProcessDefinition(null, pdId, "f86cc22d-9b50
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendProcessInstanceSignal

Description:

Method	*Return values*	*Description*
sendProcessInstanceSignal(String token, long processInstanceId,		*Send a signal to a*

String transitionName)	***ProcessInstance***	*process intance id.*

The parameter processInstanceId value is a valid workflow process instance.

The parameter uuid can be any document, mail, folder or record UUID.

The parameter transactionName is the choosen transaction.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long pId = 1046; // Some valid process instance id
            OKMWorkflow.getInstance().sendProcessInstanceSignal(null, pId, "jump2");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

endProcessInstance

Description:

Method	*Return values*	*Description*
endProcessInstance(String token, long processInstanceId)	***void***	*Ends a process intance id.*

The parameter processInstanceId value is a valid workflow process instance.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long pId = 1046; // Some valid process instance id
            OKMWorkflow.getInstance().endProcessInstance(null, pId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

deleteProcessInstance

Description:

Method	*Return values*	*Description*
deleteProcessInstance(String token, long processInstanceId)	***void***	*Delete a process intance id.*
The parameter processInstanceId value is a valid workflow process instance.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long pId = 1046; // Some valid process instance id
            OKMWorkflow.getInstance().deleteProcessInstance(null, pId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findProcessInstances

Description:

Method	*Return values*	*Description*
findProcessInstances(String token, long processDefinitionId)	***List<ProcessInstance>***	*Retrieves a list of all process instances of some registered workflows definition.*
The parameter processDefinitionId value is a valid workflow process definition.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
```

```
        try {
            // Get all workflow definitions
            for (ProcessDefinition pd : OKMWorkflow.getInstance().findAllProcessDefin
                System.out.println("WF definition: "+pd);

                // Get all process of some workflow definition
                for (ProcessInstance pi : OKMWorkflow.getInstance().findProcessInstan
                    System.out.println("PI: "+pi);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findAllProcessDefinitions

Description:

Method	*Return values*	*Description*
findAllProcessDefinitions(String token, long processDefinitionId)	***List<ProcessDefinition>***	*Retrieves a list of all registered workflows definitions.*

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;

public class Test {
    public static void main(String[] args) {
        try {
            // Get all workflow definitions
            for (ProcessDefinition pd : OKMWorkflow.getInstance().findAllProcessDefin
                System.out.println("WF definition: "+pd);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findLatestProcessDefinitions

Description:

Method	*Return values*	*Description*
findLatestProcessDefinitions(String token)	***List<ProcessDefinition>***	*Retrieves a list of the last workflows definitions.*

Several versions of the same workflow can be registered.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;

public class Test {
    public static void main(String[] args) {
        try {
            // Get all workflow definitions
            for (ProcessDefinition pd : OKMWorkflow.getInstance().findLatestProcessDe
                System.out.println("WF definition: "+pd);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findLastProcessDefinition

Description:

Method	*Return values*	*Description*
findLastProcessDefinition(String token, String name)	***ProcessDefinition***	*Retrieves last workflow definition of some specific workflow.*

The parameter name identifies an specific workflow definitions group.

Several workflow definition versions that have the same name.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;

public class Test {
    public static void main(String[] args) {
        try {
            ProcessDefinition pd = OKMWorkflow.getInstance().findLastProcessDefinitio
            System.out.println("WF definition: " + pd);
        } catch (Exception e) {
            e.printStackTrace();
```

```
        }
    }
}
```

findAllProcessDefinitionVersions

Description:

Method	*Return values*	*Description*
findAllProcessDefinitionVersions(String token, String name)	***ProcessDefinition***	*Retrieves all workflow definition versions of some specific workflow.*
The parameter name identifies an specific workflow definitions group. *Several workflow definition versions that have the same name.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessDefinition;

public class Test {
    public static void main(String[] args) {
        try {
            for (ProcessDefinition pd : OKMWorkflow.getInstance().findAllProcessDefin
                System.out.println("WF definition: " + pd);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getProcessInstance

Description:

Method	*Return values*	*Description*
getProcessInstance(String token, long processInstanceId)	***ProcessInstance***	*Returns the process instance.*
The parameter processInstanceId is a valid process instance id.		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long piId = 8108; // Some valid process instance id
            ProcessInstance pi = OKMWorkflow.getInstance().getProcessInstance(null, p
            System.out.println("PI: " + pi);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

suspendProcessInstance

Description:

Method	*Return values*	*Description*
suspendProcessInstance(String token, long processInstanceId)	***void***	*Suspends the process instance.*
The parameter processInstanceId is a valid process instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long piId = 8108; // Some valid process instance id
            OKMWorkflow.getInstance().suspendProcessInstance(null, piId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

resumeProcessInstance

Description:

Method	*Return values*	*Description*
resumeProcessInstance(String token, long processInstanceId)	***void***	*Resume the process instance.*

The parameter processInstanceId is a valid process instance id.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long piId = 8108; // Some valid process instance id
            OKMWorkflow.getInstance().resumeProcessInstance(null, piId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addProcessInstanceVariable

Description:

Method	*Return values*	*Description*
addProcessInstanceVariable(String token, long processInstanceId, String name, Object value)	***void***	*Add a variable in a process instance.*
The parameter processInstanceId is a valid process instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long piId = 8108; // Some valid process instance id
            OKMWorkflow.getInstance().addProcessInstanceVariable(null, piId, "key", "
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteProcessInstanceVariable

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
deleteProcessInstanceVariable(String token, long processInstanceId, String name)	***void***	*Delete a variable in a process instance.*
The parameter processInstanceId is a valid process instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.ProcessInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long piId = 8108; // Some valid process instance id
            OKMWorkflow.getInstance().deleteProcessInstanceVariable(null, piId, "key"
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findUserTaskInstances

Description:

Method	Return values	Description
findUserTaskInstances(String token, int offset, int limit)	***List<TaskInstance>***	*Retrieves a list of task instances assigned to the user.*

From OpenKM version 7.1.19 have been introduced the parameter offset and limit what allow to paginate the results.

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.TaskInstance;

public class Test {
    public static void main(String[] args) {
        try {
            // Get all user task intances
            for (TaskInstance ti : OKMWorkflow.getInstance().findUserTaskInstances(nu
                System.out.println(ti);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

findPooledTaskInstances

Description:

Method	*Return values*	*Description*
findPooledTaskInstances(String token, int offset, int limit)	***List<TaskInstance>***	*Retrieves a list of pooled task instances assigned to the user.*

From OpenKM version 7.1.19 have been introduced the parameter offset and limit what allow to paginate the results.

The parameter "limit" and "offset" allows you to retrieve just a portion of the results of a query.

- *The parameter "limit" is used to limit the number of results returned.*
- *The parameter "offset" is used to skip that many results before the beginning to return results.*

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.TaskInstance;

public class Test {
    public static void main(String[] args) {
        try {
            // Get all user task intances
            for (TaskInstance ti : OKMWorkflow.getInstance().findPooledTaskInstances(
                System.out.println(ti);
            }
        } catch (Exception e) {
            e.printStackTrace();
```

```
        }
    }
}
```

findTaskInstances

Description:

Method	*Return values*	*Description*
findTaskInstances(String token, long processInstanceId)	***List<TaskInstance>***	*Retrieves a list of task instances by process intance.*
The parameter processInstanceId is a valid process instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.TaskInstance;

public class Test {
    public static void main(String[] args) {
        try {
            // Get all user task intances
            long piId = 8108; // Some valid process instance id
            for (TaskInstance ti : OKMWorkflow.getInstance().findTaskInstances(null,
                System.out.println(ti);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTaskInstanceValues

Description:

Method	*Return values*	*Description*
setTaskInstanceValues(String token, long taskInstanceId, String transitionName, List<FormElement> values)	***void***	*Set a task instance values.*
The parameter taskInstanceId is a valid task instance id. *The parameter transitionName is the choosen transaction.*		

The parameter values are form element values needed for starting the workflow (not all workflow tasks need form values).

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.form.FormElement;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            List<FormElement> feList = new ArrayList<>();
            OKMWorkflow.getInstance().setTaskInstanceValues(null, tiId, "approve", fe
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addTaskInstanceComment

Description:

Method	*Return values*	*Description*
addTaskInstanceComment(String token, long taskInstanceId, String message)	***void***	*Set a task instance comment.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().addTaskInstanceComment(null, tiId, "here the co
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
    }
}
```

getTaskInstance

Description:

Method	*Return values*	*Description*
getTaskInstance(String token, long taskInstanceId)	***TaskInstance***	*Returns a task instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.TaskInstance;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            TaskInstance ti = OKMWorkflow.getInstance().getTaskInstance(null, tiId);
            System.out.println(ti);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTaskInstanceActorId

Description:

Method	*Return values*	*Description*
setTaskInstanceActorId(String token, long taskInstanceId, String actorId)	***void***	*Set the actor to a task instance.*
The parameter taskInstanceId is a valid task instance id. *The parameter actorId must be some valid OpenKM userId.*		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().setTaskInstanceActorId(null, tiId, "okmAdmin");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addTaskInstanceVariable

Description:

Method	*Return values*	*Description*
addTaskInstanceVariable(String token, long taskInstanceId, String name, Object value)	***void***	*Add variable into a task instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().addTaskInstanceVariable(null, tiId, "key", "val
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

deleteTaskInstanceVariable

Description:

Method	*Return values*	*Description*
deleteTaskInstanceVariable(String token, long taskInstanceId,	***void***	*Delete variable from a task*

String name)		*instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().deleteTaskInstanceVariable(null, tiId, "key");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

startTaskInstance

Description:

Method	*Return values*	*Description*
startTaskInstance(String token, long taskInstanceId)	***void***	*Starts a task instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().startTaskInstance(null, tiId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

endTaskInstance

Description:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Method	Return values	Description
endTaskInstance(String token, long taskInstanceId, String transitionName)	***void***	*Ends a task instance.*
The parameter taskInstanceId is a valid task instance id. *The parameter transitionName is a transaction name.*		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().endTaskInstance(null, tiId, "end");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

suspendTaskInstance

Description:

Method	Return values	Description
suspendTaskInstance(String token, long taskInstanceId)	***void***	*Suspend a task instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().suspendTaskInstance(null, tiId);
        } catch (Exception e) {
            e.printStackTrace();
        }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
}
```

resumeTaskInstance

Description:

Method	*Return values*	*Description*
resumeTaskInstance(String token, long taskInstanceId)	***void***	*Resume a task instance.*
The parameter taskInstanceId is a valid task instance id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;

public class Test {
    public static void main(String[] args) {
        try {
            long tiId = 8110; // Some valid task instance id
            OKMWorkflow.getInstance().resumeTaskInstance(null, tiId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getToken

Description:

Method	*Return values*	*Description*
getToken(String token, long tokenId)	***Token***	*Return a token object.*
The parameter tokenId is a valid token id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
```

```
        try {
            long tkId = 8110; // Some valid token id
            Token tk = OKMWorkflow.getInstance().getToken(null, tkId);
            System.out.println(tk);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

addTokenComment

Description:

Method	*Return values*	*Description*
addTokenComment(String token, long tokenId, String message)	***void***	*Add a token comment.*
The parameter tokenId is a valid token id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            Token tk = OKMWorkflow.getInstance().addTokenComment(null, tkId, "here th
            System.out.println(tk);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

suspendToken

Description:

Method	*Return values*	*Description*
suspendToken(String token, long tokenId)	***void***	*Suspend a token.*
The parameter tokenId is a valid token id.		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            Token tk = OKMWorkflow.getInstance().suspendToken(null, tkId);
            System.out.println(tk);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

resumeToken

Description:

Method	*Return values*	*Description*
resumeToken(String token, long tokenId)	***void***	*Resume a token.*
The parameter tokenId is a valid token id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            Token tk = OKMWorkflow.getInstance().resumeToken(null, tkId);
            System.out.println(tk);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendTokenSignal

Description:

Method	*Return values*	*Description*
sendTokenSignal(String token, long tokenId, String transitionName)	***Token***	*Send a signal.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

The parameter tokenId is a valid token id.

The parameter transitionName is a transaction name.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            Token tk = OKMWorkflow.getInstance().sendTokenSignal(null, tkId, "end");
            System.out.println(tk);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

setTokenNode

Description:

Method	*Return values*	*Description*
setTokenNode(String token, long tokenId, String nodeName)	***void***	*Set token node.*

The parameter tokenId is a valid token id.

The parameter nodeName is a node name.

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            OKMWorkflow.getInstance().setTokenNode(null, tkId, "formA");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
}
```

endToken

Description:

Method	*Return values*	*Description*
endToken(String token, long tokenId)	***Token***	*End token.*
The parameter tokenId is a valid token id.		

Example:

```
package com.openkm;

import com.openkm.api.OKMWorkflow;
import com.openkm.bean.workflow.Token;

public class Test {
    public static void main(String[] args) {
        try {
            long tkId = 8110; // Some valid token id
            OKMWorkflow.getInstance().setTokenNode(null, tkId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Commons classes

OpenKM commons classes:

OpenKM common class	*Description*
ArchiveUtils	*Utility methods for archive handling. For example, to create a ZIP or JAR.*
ExecutionUtils	*Execution related methods. For example, to run a script or JAR.*
FileLogger	*Utility methods to generate a log. For example, to show error, info or warning.*
FileUtils	*File related utility methods. For example, to get the filename or extension.*
FormatUtil	*Utility methods used by formatters. For example, to format a file size or format time.*
GeneralUtils	*Generic utility methods. For example, to notify something to an user.*
ISO8601	*Utility class what provides bidirectional format transformation between java time objects and ISO8601 format used for metadata date fields.*
MailUtils	*Utility methods to send a mail. For example to send message or send document.*
PathUtils	*Utility methods to handle paths. For example, get the name or get the parent path.*
PDFUtils	*Utility methods to convert PDF files. For example, to stamp image or text.*
PrincipalUtils	*Utility methods for retrieving information from the security context. For example, get the actual userId from the security context session or get the actual tenant of the logged user.*
StackTraceUtils	*Utility method for the stack.*
TemplateUtils	*Utility methods to handle templates. For example, replace a text in a template.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

WorkflowUtils	*Utility methods to handle workflows.*
ShardUtils	*Utility methods to have a repository distributed on several disks*
LegacySrv	*Utility methods to perform SQL query and HQL queries.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ArchiveUtils

Utility methods for archive handling. For example, to create a ZIP or JAR.

Methods

createZip

Description:

Method	Return values	Description
createZip(File path, OutputStream os)	**void**	*Create ZIP archive from file.*
***path**: The file to compress.* ***os**: The output stream of the file.*		

Example:

```
package com.openkm;

import java.io.ByteArrayOutputStream;
import java.io.File;

import com.openkm.util.ArchiveUtils;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            ArchiveUtils archiveUtils = ContextWrapper.getContext().getBean(ArchiveUt
            File file = new File("/home/openkm/test.pdf");
            ByteArrayOutputStream baos = new ByteArrayOutputStream();
            archiveUtils.createZip(file, baos);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createJar

Description:

Method	Return values	Description
createJar(File path, String root, OutputStream os)	**void**	*Recursively create JAR archive from directory.*

***path**: The file to compress.*

***root**: A text with destiny of the file.*

***os**: The output stream of the file.*

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileOutputStream;
import java.io.StringWriter;

import com.openkm.util.ArchiveUtils;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.PathUtils;
import com.openkm.util.impexp.RepositoryExporter;
import com.openkm.util.impexp.TextInfoDecorator;

public class Test {

    public static void main(String[] args) {
        try {
            ArchiveUtils archiveUtils = ContextWrapper.getContext().getBean(ArchiveUt
            PathUtils pathUtils = ContextWrapper.getContext().getBean(PathUtils.class
            FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class
            RepositoryExporter repositoryExporter = ContextWrapper.getContext().getBe

            StringWriter out = new StringWriter();

            File dirFile = new File("/home/openkm");
            final File tmpFile = File.createTempFile("test", ".jar", dirFile);
            FileOutputStream os = new FileOutputStream(tmpFile);

            File tmp = fileUtils.createTempDir();
            repositoryExporter.exportFolder(null, "/okm:root/jar", tmp, false, out, n

            // Jar files
            archiveUtils.createJar(tmp, pathUtils.getName("/home/openkm"), os);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ExecutionUtils

Execution related methods. For example, to run a script or JAR..

Methods

runScript

Description:

Method	*Return values*	*Description*
runScript(File script)	***Object[]***	*Execute script from file.*
***script**: The name of the file to run.*		

Example:

```
package com.openkm;

import java.io.File;

import com.openkm.core.Config;
import com.openkm.util.ExecutionUtils;

public class Test {

    public static void main(String[] args) {
        try {
            File script = new File("/home/openkm/test.bsh");
            ExecutionUtils.runScript(script);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

runJar

Description:

Method	*Return values*	*Description*
runJar(File jar)	***Object***	*Execute jar from file.*
***jar**: The name of the file to run.*		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import java.io.File;

import com.openkm.util.ExecutionUtils;

public class Test {

    public static void main(String[] args) {
        try {
            File jar = new File("/home/openkm/test.jar");
            ExecutionUtils.getInstance().runJar(jar);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

runCmd

Description:

Method	*Return values*	*Description*
runCmd(String cmd)	***CmdExecutionResult***	*Execute command line.*
***cmd**: The command line to execute.*		

Example:

```
package com.openkm;

import com.openkm.util.ExecutionUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ExecutionUtils.runCmd("mkdir /home/openkm/test");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FileLogger

Utility methods to generate a log. For example, to show error, info or warning.

Methods

info

Description:

Method	*Return values*	*Description*
info(String baseName, String message, Object... params)	***void***	*Static file logger with info.*
***baseName**: The name of file to write.* ***message**: A text with message for info.*		

Example:

```
package com.openkm;

import com.openkm.util.FileLogger;

public class Test {

    public static void main(String[] args) {
        try {
            FileLogger.info("test.log", "This is info");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

warn

Description:

Method	*Return values*	*Description*
warn(String baseName, String message, Object... params)	***void***	*Static file logger with warn.*
***baseName**: The name of file to write.* ***message**: A text with message for warning.*		

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import com.openkm.util.FileLogger;

public class Test {

    public static void main(String[] args) {
        try {
            FileLogger.warn("test.log", "This is warning");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

error

Description:

Method	*Return values*	*Description*
error(String baseName, String message, Object... params)	***void***	*Static file logger with error.*
***baseName**: The name of file to write.* ***message**: A text with message for error.*		

Example:

```
package com.openkm;

import com.openkm.util.FileLogger;

public class Test {

    public static void main(String[] args) {
        try {
            FileLogger.error("test.log", "This is error");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FileUtils

File related utility methods. For example, to get the filename or extension.

Methods

getFileName

Description:

Method	*Return values*	*Description*
getFileName(String file)	***String***	*Returns the name of the file without the extension.*
***file**: The full name of the file.*		

Example:

```
package com.openkm;

import java.io.File;

import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        try {
            FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class
            File file = new File("/home/openkm/test.png");
            String docName = PathUtils.getName(file.getPath());
            String baseName = fileUtils.getFileName(docName);
            System.out.println(baseName);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getFileExtension

Description:

Method	*Return values*	*Description*
getFileExtension(String file)	***String***	*Returns the filename extension.*

***file**: The full name of the file.*

Example:

```
package com.openkm;
import java.io.File;

import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        try {
            FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class
            File file = new File("/home/openkm/test.png");
            String docName = PathUtils.getName(file.getPath());
            String extension = fileUtils.getFileExtension(docName);
            System.out.println(extension);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

FormatUtil

Utility methods used by formatters. For example, to format a file size or format time.

Methods

formatSize

Description:

Method	*Return values*	*Description*
formatSize(long bytes)	***String***	*Format the document size for human readers.*
***bytes**: The size of the file content.*		

Example:

```
package com.openkm;

import java.io.File;
import com.openkm.util.FormatUtil;

public class Test {

    public static void main(String[] args) {
        try {
            File file = new File("/home/openkm/test.png");
            System.out.println(FormatUtil.formatSize(file.length()));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

parseSize

Description:

Method	*Return values*	*Description*
parseSize(String text)	***long***	*Parse human-readable sizes.*
***text**: The size to parse.*		

Example:

```
package com.openkm;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.util.FormatUtil;

public class Test {
    public static void main(String[] args) {
        try {
         System.out.println(FormatUtil.parseSize("1024.7"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

formatDate

Description:

Method	*Return values*	*Description*
formatDate(Calendar cal)	***String***	*Format calendar date.*
***cal**: The date to format.*		

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.util.FormatUtil;

public class Test {
    public static void main(String[] args) {
        try {
         System.out.println(FormatUtil.formatDate(Calendar.getInstance()));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ISO8601

Utility class what provides bidirectional format transformation between java time objects and ISO8601 format used for metadata date fields.

Methods

parseBasic

Description:

Method	*Return values*	*Description*
parseBasic(String value)	*String*	*Return date from an ISO8601 basic format string.*
***value**: A ISO86101 basic format string value.*		

Example:

```
package com.openkm;

import java.text.SimpleDateFormat;
import java.util.Calendar;

import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");
            Calendar cal = ISO8601.parseBasic("20180222000000");
            System.out.println(sdf.format(cal.getTime()));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

formatBasic

Description:

Method	*Return values*	*Description*
formatBasic(Calendar calendar)	*Calendar*	*Return ISO8601 basic format from date.*
***calendar**: The date to format.*		

Example:

```
package com.openkm;

import java.util.Calendar;

import com.openkm.util.ISO8601;

public class Test {

    public static void main(String[] args) {
        try {
            Calendar cal = Calendar.getInstance();
            System.out.println(ISO8601.formatBasic(cal.getTime()));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

GeneralUtils

Generic utility methods. For example, to notify something to an user.

Methods

uiNotify

Description:

Method	*Return values*	*Description*
uiNotify(String msg)	***void***	*Notify to current authenticated user.*
***msg**: A text with the message of notify.*		

The user must be logged before executing the methods.

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.GeneralUtils;

public class Test {

    public static void main(String[] args) {
        try {
            GeneralUtils generalUtils = ContextWrapper.getContext().getBean(GeneralUt
            generalUtils.uiNotify("Message test!");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

MailUtils

Utility methods to send a mail. For example to send message or send document.

Methods

sendMessage

Description:

Method	*Return values*	*Description*
sendMessage(String toAddress, String subject, String content)	***void***	*Send mail without FROM addresses.*
***toAddress**: The destination addresses for mail.* ***subject**: The mail subject.* ***content**: The mail body.*		

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.MailUtils;

public class Test {

    public static void main(String[] args) {
        try {
            MailUtils mailUtils = ContextWrapper.getContext().getBean(MailUtils.class
            mailUtils.sendMessage("gnu.java.sergio@gmail.com", "this is the subject",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendMessage

Description:

Method	*Return values*	*Description*
sendMessage(String fromAddress, String toAddress, String subject, String content)	***void***	*Send mail with FROM addresses.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

fromAddress: *The from addresses for mail.*

toAddress*: The destination addresses for mail.*

subject*: The mail subject.*

content*: The mail body.*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.MailUtils;

public class Test {

    public static void main(String[] args) {
        try {
            MailUtils mailUtils = ContextWrapper.getContext().getBean(MailUtils.class
            mailUtils.sendMessage("from_noreply@openkm.com", "to_noreply@openkm.com",
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

sendDocuments

Description:

Method	*Return values*	*Description*
sendDocuments(String fromAddress, List<String> toAddresses, String subject, String text, List<String> docsPath)	***MimeMessage***	*Send document to non-registered OpenKM users.*

fromAddress: *The from addresses for mail.*

toAddress*: The list of destination addresses for mail.*

subject*: The mail subject.*

text*: The mail body.*

docsPath*: The path of the documents to be send.*

Example:

```
package com.openkm;

import java.util.ArrayList;
import java.util.List;

import com.openkm.util.ContextWrapper;
import com.openkm.util.MailUtils;

public class Test {

    public static void main(String[] args) {
        try {
            MailUtils mailUtils = ContextWrapper.getContext().getBean(MailUtils.class
            List<String> docsPath = new ArrayList<>();
            List<String> listAddress = new ArrayList<>();
            listAddress.add("to_noreply@openkm.com");
            docsPath.add("/home/openkm/test.png");
            docsPath.add("/home/openkm/test2.png");
            docsPath.add("/home/openkm/test3.png");
            mailUtils.sendDocuments("from_noreply@openkm.com", listAddress, "this is
                    docsPath);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

PathUtils

Utility methods to handle paths. For example, get the name or get the parent path.

Methods

getParent

Description:

Method	*Return values*	*Description*
getParent(String path)	***String***	*Get parent node.*
***path**: The path of the file.*		

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        try {
            System.out.println(PathUtils.getParent("/okm:root/logo.png"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getName

Description:

Method	*Return values*	*Description*
getName(String path)	***String***	*Get name node.*
***path**: The path of the file.*		

Example:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        try {
            System.out.println(PathUtils.getName("/okm:root/logo.png"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getContext

Description:

Method	*Return values*	*Description*
getContext(String path)	***String***	*Get path context. For example "/okm:root/test.txt" -> "/okm:root".*
***path**: The path of the file.*		

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        try {
            System.out.println(PathUtils.getContext("/okm:root/logo.png"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

PDFUtils

Utility methods to convert PDF files. For example, to stamp image or text.

Methods

merge

Description:

Method	*Return values*	*Description*
merge(List<InputStream> inputs, OutputStream output)	***void***	*Merge several PDFs into a new one.*
***inputs**: List of inputs stream for merge.* ***output**: Output stream of the file.*		

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;

import org.apache.commons.io.IOUtils;

import com.openkm.api.OKMDocument;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.PDFUtils;
import com.openkm.util.PathUtils;

public class Test {

    public static void main(String[] args) {
        FileOutputStream fos = null;
        FileInputStream fis = null;
        File tmp = null;
        FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class);

        try {
            OKMDocument okmDocument = ContextWrapper.getContext().getBean(OKMDocument
            PDFUtils pdfUtils = ContextWrapper.getContext().getBean(PDFUtils.class);
            PathUtils pathUtils = ContextWrapper.getContext().getBean(PathUtils.class

            tmp = fileUtils.createTempFile("pdf");
            fos = new FileOutputStream(tmp);

            List<InputStream> inputs = new ArrayList<>();
```

```
            inputs.add(okmDocument.getContent(null, "c0984ecf-8d9e-40f8-b5fb-2c83d202
            inputs.add(okmDocument.getContent(null, "b3d0af95-076c-4cc4-9b85-bd3e7eb2

            // Merge document
            pdfUtils.merge(inputs, fos);

            // Create document in repository
            String fldPath = pathUtils.getParent("/okm:root/test3.pdf"); // all docum
            String docPath = fldPath + "/docMerge.pdf";
            fis = new FileInputStream(tmp);
            okmDocument.createSimple(null, docPath, fis);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            IOUtils.closeQuietly(fis);
            IOUtils.closeQuietly(fos);
            fileUtils.deleteQuietly(tmp);
        }
    }
}
```

stampImage

Description:

Method	*Return values*	*Description*
stampImage(InputStream input, byte[] image, OutputStream output)	***void***	*Stamp PDF document with image watermark.*
***input**: The input stream of the file.* ***image**: The content of the file.* ***output**: Output stream of the file.*		

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.nio.file.Files;
import java.nio.file.Paths;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PDFUtils;

public class Test {

    public static void main(String[] args) {
        try {
```

```
            PDFUtils pdfUtils = ContextWrapper.getContext().getBean(PDFUtils.class);

            String path = "/home/openkm/test.pdf";
            File file = new File(path);
            InputStream is = new FileInputStream(file);
            OutputStream out = new FileOutputStream(file);
            byte[] data = Files.readAllBytes(Paths.get("/home/luis/test.png"));
            pdfUtils.stampImage(is, data, out);
            is.close();
            out.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

stampText

Description:

Method	*Return values*	*Description*
stampText(InputStream input, String text, OutputStream output)	***void***	*Stamp PDF document with text watermark.*

***input**: The input stream of the file.*

***text:** A text for the stamp.*

***output**: Output stream of the file.*

Example:

```
package com.openkm;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PDFUtils;

public class Test {

    public static void main(String[] args) {
        try {
            PDFUtils pdfUtils = ContextWrapper.getContext().getBean(PDFUtils.class);

            String path = "/home/openkm/test.pdf";
            File file = new File(path);
            InputStream is = new FileInputStream(file);
            OutputStream out = new FileOutputStream(file);
            pdfUtils.stampText(is, "This is the text", out);
```

```
            is.close();
            out.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getNumberOfPages

Description:

Method	*Return values*	*Description*
getNumberOfPages(String token, String docId)	***int***	*Return the number of pages for a PDF.*
***input**: The input stream of the file.* ***text:** A text for the stamp.* ***docId**: PDF OpenKM unique identifier.*		

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.PDFUtils;

public class Test {

    public static void main(String[] args) {
        try {
            PDFUtils pdfUtils = ContextWrapper.getContext().getBean(PDFUtils.class);
            String docId = "e6b61e8d-4bf1-4c37-828f-09ecc34261cc";
            System.out.println(pdfUtils.getNumberOfPages(null, docId));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

PrincipalUtils

Utility methods for retrieving information from the security context. For example, get the actual userId from the security context session or get the actual tenant of the logged user.

Methods

getUser

Description:

Method	*Return values*	*Description*
getUser()	***String***	*Get the userId of the the logged user.*

The user must be logged before executing the methods.

Example:

```
package com.openkm;

import com.openkm.principal.PrincipalUtils;

public class Test {

    public static void main(String[] args) {
        try {
            System.out.println(PrincipalUtils.getUser());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getTenantId

Description:

Method	*Return values*	*Description*
getTenantId()	***long***	*Get the tenantId of the logged user.*

The user must be logged before executing the methods.

Example:

```
package com.openkm;

import com.openkm.principal.PrincipalUtils;

public class Test {

    public static void main(String[] args) {
        try {
            long tenantId = PrincipalUtils.getTenantId();
            System.out.println(tenantId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

StackTraceUtils

Utility methods for the stack.

Methods

isCallingMe

Description:

Method	Return values	Description
isCallingMe(String className)	**Boolean**	*Return true when a className is in the current stack*

Example:

```
package com.openkm;

package com.openkm.test;

import com.openkm.spring.PrincipalUtils;

public class Test {
 public static void main(String[] args) throws Exception {
  System.out.println(StackTraceUtilsUtils.isCallingMe("com.openkm.automation.action.U
 }
}
```

toString

Description:

Method	Return values	Description
toString(Throwable t)	**String**	*Return exception stack as standard text.*

Example:

```
package com.openkm;

package com.openkm.test;

import com.openkm.spring.PrincipalUtils;

public class Test {
 public static void main(String[] args) throws Exception {
        try {
            // Force exception
            List keys = null;
   for (String key : keys) {
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
    }
          } catch (Exception e) {
    System.out.println(StackTraceUtilsUtils.toString(e));
   }
  }
 }
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

TemplateUtils

Utility methods to handle templates. For example, replace a text in a template.

Methods

templateExists

Description:

Method	*Return values*	*Description*
templateExists(String name)	***boolean***	*Check for template existence.*
name: *The name of the template.*		

Example:

```
package com.openkm;

import com.openkm.util.TemplateUtils;

public class Test {

    public static void main(String[] args) {
        try {
            System.out.println("Exist template: " + TemplateUtils.templateExists("Ope
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

replace

Description:

Method	*Return values*	*Description*
replace(String name, String template, Map<String, Object> model)	***String***	*Quick replace utility function.*
name: *The name of the template.* ***template****: A text whith the template definition.* ***model****: A map use to reaplace elements in the template.*		

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Example:

```
package com.openkm;

import java.io.File;
import java.util.HashMap;
import java.util.Map;

import com.openkm.core.Config;
import com.openkm.util.ContextWrapper;
import com.openkm.util.FileUtils;
import com.openkm.util.TemplateUtils;

public class Test {

    public static void main(String[] args) {
        try {
            FileUtils fileUtils = ContextWrapper.getContext().getBean(FileUtils.class

            Map<String, Object> hm = new HashMap<>();
            File tmpFileIn = fileUtils.createTempFileFromMime("text/plain");
            hm.put("fileIn", tmpFileIn.getPath());
            String cmd = TemplateUtils.replace("SYSTEM_XLS2CSV", Config.SYSTEM_CATDOC_
            System.out.println("cmd template: " + cmd);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

WorkflowUtils

Utility methods to handle workflows.

Methods

findProcessInstancesByNode

Description:

Method	*Return values*	*Description*
findProcessInstancesByNode(String uuid)	***List<ProcessInstance>***	*Retrieves a list of all process instances of a node*
***uuid** string type is the uuid of the document, folder, mail or record.*		

Example:

```
package com.openkm;

import com.openkm.bean.workflow.ProcessInstance;
import com.openkm.util.WorkflowUtils;

public class Test {

    public static void main(String[] args) {
        try {
            String uuid = "some node uuid";
            for (ProcessInstance processInstance : WorkflowUtils.findProcessInstances
                System.out.println(processInstance);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

ShardUtils

Utility methods to have a repository distributed on several disks

Methods

setCurrentShard

Description:

Method	*Return values*	*Description*
setCurrentShard(Long shardId)	***void***	*Set the current shard.*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            Long shardId = new Long(1);
            shardUtils.setCurrentShard(shardId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

createShardDirs

Description:

Method	*Return values*	*Description*
createShardDirs(Long shardId)	***void***	*Create new shard folder*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {
```

```
    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            Long shardId = new Long(1);
            shardUtils.createShardDirs(shardId);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

changeNodeShard

Description:

Method	*Return values*	*Description*
changeNodeShard(String uuid, Long newShard)	***void***	*Change shard of a node.*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            Long newShardId = new Long(2);
            shardUtils.changeNodeShard("e2391b01-ce10-42c7-94a9-b0d6b394048e", newSha
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getShardIds

Description:

Method	*Return values*	*Description*
getShardIds()	***List<Long>***	*Get the shard ids.*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            for (long id : shardUtils.getShardIds()) {
                System.out.println(id);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getShardSize

Description:

Method	*Return values*	*Description*
getShardSize(Long shardId)	***Long***	*Get the size of a shard.*

Example:

```
package com.openkm;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            Long shardId = new Long(1);
            System.out.println(shardUtils.getShardSize(shardId));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getShardPaths

Description:

Method	*Return values*	*Description*
getShardPaths(Long shardId)	***List<String>***	*Get the path of a shard.*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            Long shardId = new Long(1);
            List<String> list = shardUtils.getShardPaths(shardId);
            for (String path : list) {
                System.out.println(path);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getShards

Description:

Method	*Return values*	*Description*
getShards()	***List<Shard>***	*Get the shards*

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.db.bean.Shard;
import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            List<Shard> list = shardUtils.getShards();
            for (Shard shard : list) {
                System.out.println(shard);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

getShardFolders

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Description:

Method	*Return values*	*Description*
getShardFolders()	***List<File>***	*Get shard folders.*

Example:

```
package com.openkm;

import java.io.File;
import java.util.List;

import com.openkm.util.ContextWrapper;
import com.openkm.util.ShardUtils;

public class Test {

    public static void main(String[] args) {
        try {
            ShardUtils shardUtils = ContextWrapper.getContext().getBean(ShardUtils.cl
            List<File> list = shardUtils.getShardFolders();
            for (File file : list) {
                System.out.println(file.getPath());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

LegacySrv

executeSQL

Description:

Method	*Return values*	*Description*
executeSQL(String sql)	***List<List<String>>***	*Executes SQL script.*

```
SELECT NBS_UUID, NBS_AUTHOR, NBS_NAME, NBS_NOTES FROM OKM_NODE_BASE LIMIT 10;
```

The parameter named sql only allows a single SQL script.

This action can only be done by users with administrator grants.

Example:

```
package com.openkm;

import java.util.List;

import com.openkm.db.service.LegacySrv;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            LegacySrv legacySrv = ContextWrapper.getContext().getBean(LegacySrv.class
            List<List<String>> results = legacySrv.executeSQL("SELECT NBS_UUID, NBS_A
            for (List<String> row : results) {
                System.out.println("uuid: " + row.get(0));
                System.out.println("Author: " + row.get(1));
                System.out.println("Name: " + row.get(2));
                System.out.println("Notes: " + row.get(3));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

executeHQL

Description:

Method	*Return values*	*Description*
executeHQL(String hql)	***List<Object>***	*Executes HQL script.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
SELECT uuid, name from NodeBase where name = 'okm:root';
```

The parameter named hql only allows a single HQL script.

This action can only be done by users with administrator grants.

Example:

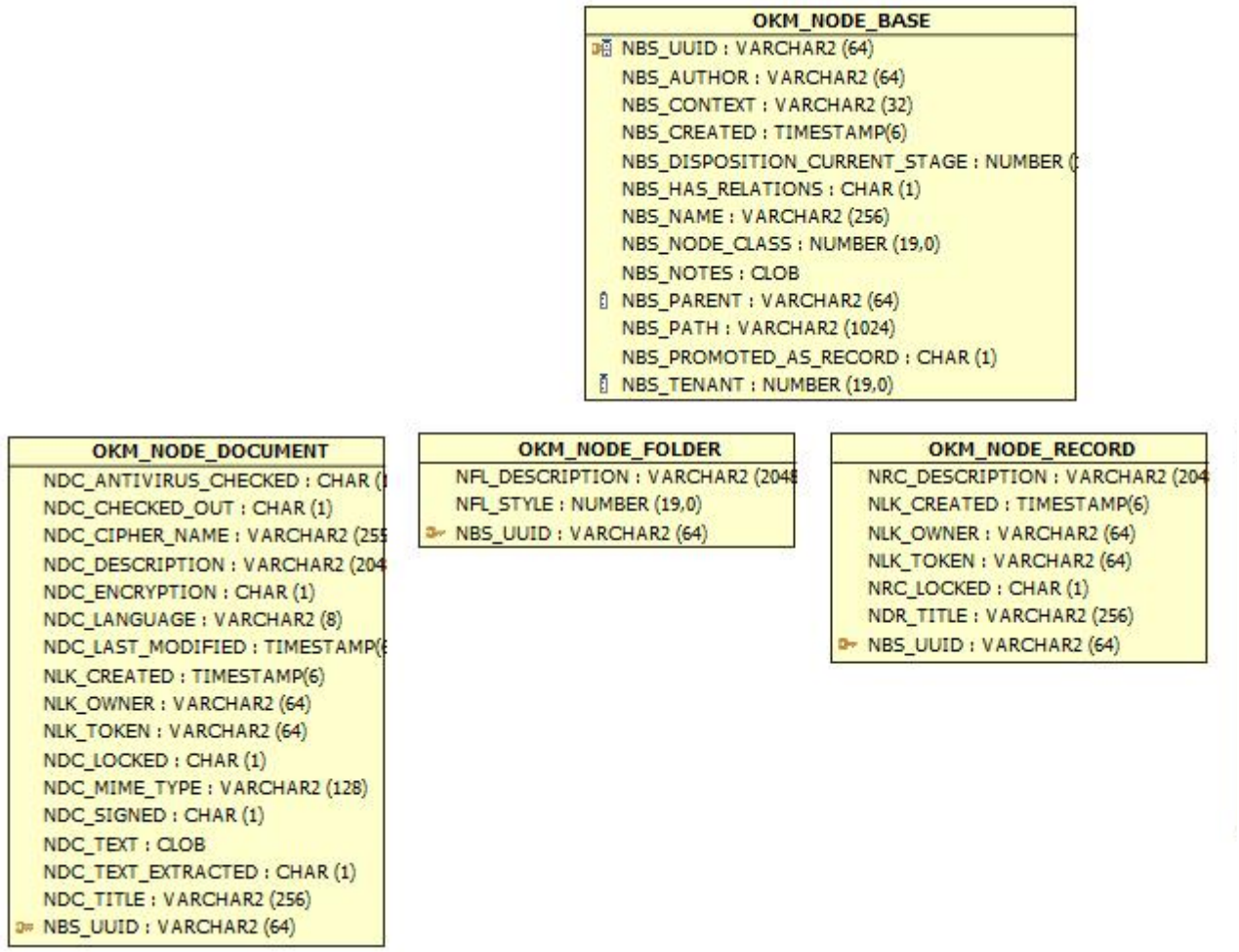
```
package com.openkm;

import java.util.List;

import com.openkm.db.service.LegacySrv;
import com.openkm.util.ContextWrapper;

public class Test {

    public static void main(String[] args) {
        try {
            LegacySrv legacySrv = ContextWrapper.getContext().getBean(LegacySrv.class
            List<Object> results = legacySrv.executeHQL("SELECT * from NodeBase nb wh
            for (Object obj : results) {
                if (obj instanceof Object[]) {
                    Object[] ao = (Object[]) obj;

                    for (Object o : ao) {
                        System.out.println(String.valueOf(o));
                    }
                } else {
                    System.out.println(String.valueOf(obj));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Database description

Node types

These tables store the information about a node in OpenKM.



*The table named **OKM_NODE_BASE** store the common fields (name, author, creation date, among others) for all the node types (Document, Folder, Mail or Record).*

*The field named **NBS_UUID** is the **unique node ID**. **NBS_UUID FIELD** is the **foreight key between parent table** (OKM_NODE_BASE) **and childs** (OKM_NODE_DOCUMENT, OKM_NODE_FOLDER, OKM_NODE_MAIL, OKM_NODE_RECORD).*

There are other useful tables:

- ***OKM_NODE_DOCUMENT**: Store information about documents.*
- ***OKM_NODE_FOLDER**: Store information about folders.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- ***OKM_NODE_MAIL**: Store information about mails.*
- ***OKM_NODE_RECORD**: Store information about records.*

SQL example queries

Query used to retrieve the information about every document:

```
select * from okm_node_base nb
  inner join okm_node_document nd on nd.nbs_uuid = nb.nbs_uuid
```

Query to get records what the name starts with 'test':

```
select * from okm_node_base nb
  inner join okm_node_record nr on nr.nbs_uuid = nb.nbs_uuid
  where nb.nbs_name like 'test%'
```

To find a Document by uuid could use this query:

```
select * from okm_node_base nb
  inner join okm_node_document nd on nd.nbs_uuid = nb.nbs_uuid
  where nb.nbs_uuid = 'fc35ce80-57f2-4026-ae77-7a11bca042ed'
```

List of child nodes with standard security

```
SELECT NB.NBS_UUID FROM OKM_NODE_BASE NB left outer join OKM_NODE_USER_PERMISSION UP
```

Mark documents to being indexed again

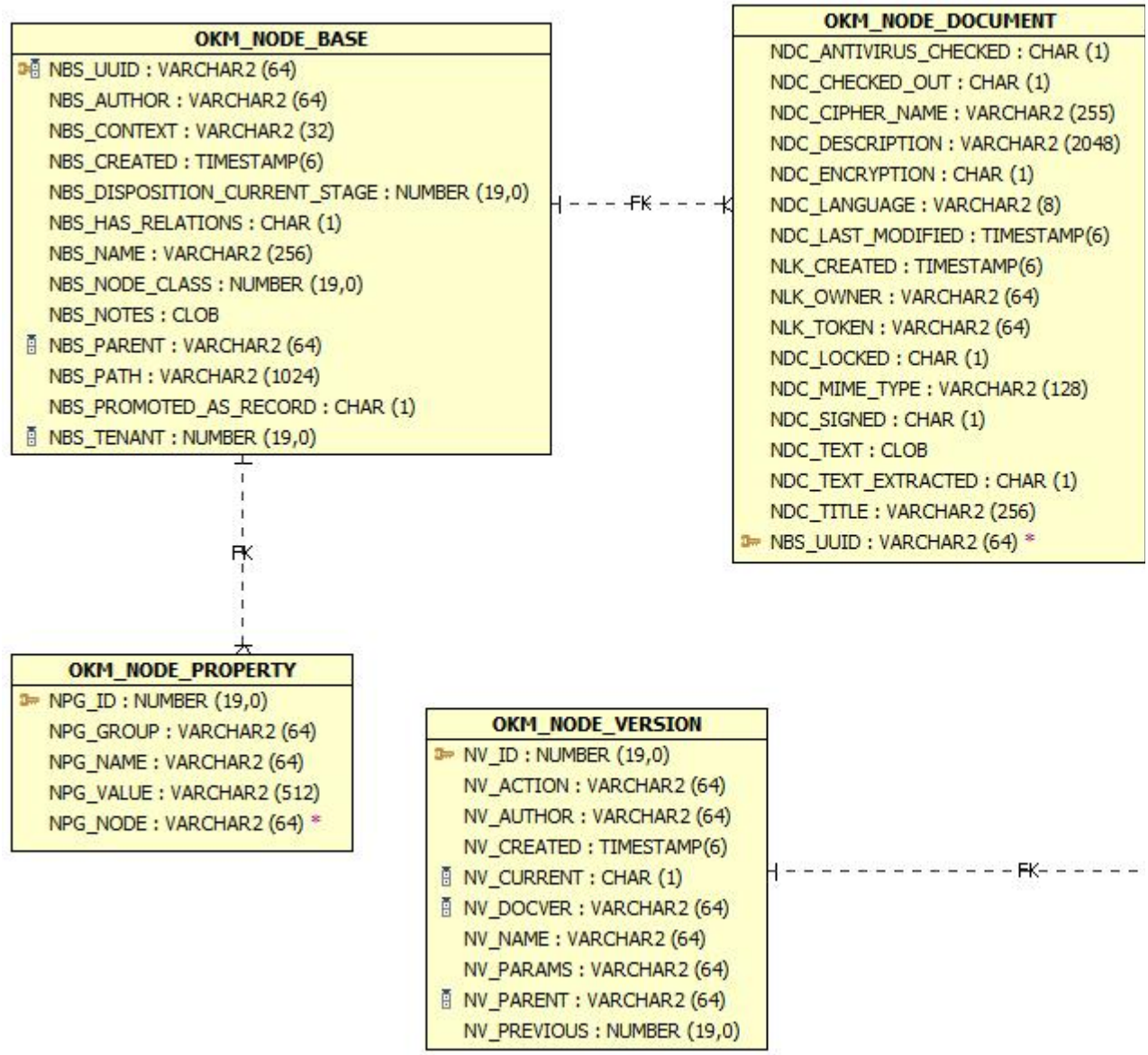
The column named NDC_TEXT_EXTRACTED in the table OKM_NODE_DOCUMENT it takes control of what documents has been processed into the Text extraction queue and what are still pending to be procesed (value 'T' means processed and 'F' means pending). If you want to reindex the whole repository again or some files you might execute a query like this one:

```
update OKM_NODE_DOCUMENT set NDC_TEXT_EXTRACTED='F'
```

Node version and properties

These tables store information about node version history:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*The table named **OKM_NODE_VERSION** store version information about every version of each node.*

- *The field **NV_ID** is the id of a version.*
- *The value of the field named **NV_PARENT** is the **unique document ID** (the same value as NBS_UUID).*

*The table named **OKM_NODE_DOCUMENT_VERSION** store the information of all versions of a document.*

- *The field named **NDV_UUID** is the **unique document version ID**.*
- *The value of the field named **NDV_PARENT** is the **unique document ID** (the same value as NBS_UUID).*

*In the table named **OKM_NODE_PROPERTY** are stored the metadata of a node (Document, Folder, Mail, Record).*

- *The value of the field named **NPG_NODE** value is the **unique document ID** (the same value as NBS_UUID).*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *The field named **NPG_GROUP** is the **metadata group name** (for example okg:consulting)*
- *The field named **NPG_NAME** is the **metadata field** name (for example okp:consulting.comment)*

*The table named **OKM_NODE_PROPERTY_VERSION** store the information of all versions of the properties.*

- *The field named **NPV_VERSION** is the number of version of this property (the same value as NV_ID)*

SQL example queries

Query to get all changes of a document named 'test.pdf':

```
select nv.*, ndv.* from okm_node_document nd
  inner join okm_node_version nv on nv.nv_parent = nd.nbs_uuid
  inner join okm_node_base nb on nd.nbs_uuid = nb.nbs_uuid
  inner join okm_node_document_version ndv on ndv.ndv_parent = nd.nbs_uuid
  where nb.nbs_name = 'test.pdf'
```

This query prints the following info:

Column	*Type*	*Description*
NV_ACTION	*DOCUMENT_CREATED* *FOLDER_CREATED* *MAIL_CREATED* *RECORD_CREATED* *DOCUMENT_CHANGED* *DOCUMENT_IMPORTED* *FOLDER_IMPORTED* *RECORD_IMPORTED* *MAIL_IMPORTED* *PROPERTY_GROUP_ADDED* *PROPERTY_GROUP_REMOVED* *PROPERTY_GROUP_CHANGED*	*What kind of change were made.*
NV_NAME	***String***	*Version number.*
NDV_UUID	***String***	*Document content version id.*

Query to get all the metadata values from a document named 'test.pdf' and their changes:

```
select npv.* from okm_node_document nd
  inner join okm_node_version nv on nv.nv_parent = nd.nbs_uuid
  inner join okm_node_base nb on nd.nbs_uuid = nb.nbs_uuid
  inner join okm_node_property_version npv on npv.npv_version = nv.nv_id
  where nb.nbs_name = 'test.pdf'
  order by npv.npv_version desc
```

This query will print all metadata values for each version. Usually, the actual version of these metadata values are the ones having the biggest npv_version value.

The next query is useful to know where all the versions of a file named 'test.pdf' are stored into the datastore:

Only applicable to the default datastore configuration (case all the versions of the documents are stored in the file system).

Take a look at Datastore for more information.

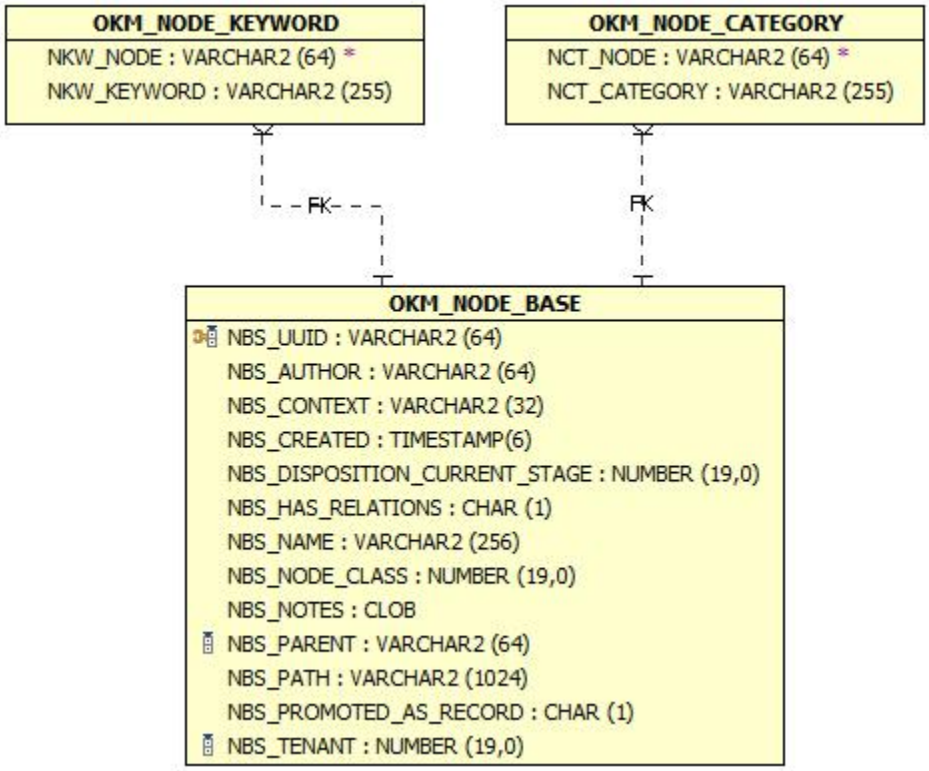
```
select ndv_uuid from okm_node_document nd
  inner join okm_node_document_version nv on nv.ndv_parent = nd.nbs_uuid
  inner join okm_node_base nb on nd.nbs_uuid = nb.nbs_uuid
  where nb.nbs_name = 'test.pdf'
```

*For example '**f0d7f0a1**-b337-4917-b592-56e163b81cde'. This means your file is located in the physical drive in the path (based in first eight characters of this id):*

*/home/user/tomcat_home/repository/datastore/**f0/d7/f0/a1**/f0d7f0a1-b337-4917-b592-56e163b81cde*

Node categories and keywords

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



*The value of the field named **NCT_NODE** is the **unique document ID** (the same value as NBS_UUID).*

*The value of the field named **NKW_NODE** is the **unique document ID** (the same value as NBS_UUID).*

SQL example queries

Query to extract all keywords associated with a node named 'test.pdf':

```
select * from okm_node_base nb
  inner join okm_node_keyword nk on nk.nkw_node = nb.nbs_uuid
  where nb.nbs_name = 'test.pdf'
```

Query to get all the categories for a node named 'test.pdf':

```
select nd.nbs_uuid, nbs_name, npg_group, npg_name, npg_value
  from okm_node_base nb
  inner join okm_node_category nc on nc.nct_node = nb.nbs_uuid
  where nb.nbs_name = 'test'
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Advanced samples

The samples below are demonstrations of how to building your own user interface based in external application.

Field / Property	*Description*
Custom UI template	*Complete wizard demonstration.* *You have a video in youtuve where it is shown how this sample it works https://www.youtube.com/watch?v=do-uVuk816M*
Form UI template	*Demonstrates how to create a document from a thirdparty application form.*
Security Provider	*Create your own security provider.*
Set unique document name based on metadata	*Create unique name and unique id field based on metadata*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Custom UI template

Developer information for Sample Spring Boot base project with OpenKM authentication and support for JSP, JSTL & REST.

To learn more about Spring Boot, please visit https://projects.spring.io/spring-boot/

There is also a very good tutorial about Spring Boot at https://spring.io/guides/gs/spring-boot/

> *You can download the application (**code-samples.x.x.zip**) from OpenKM Download center.*

General description

Properties file

*You have to create a file called **custom.properties** inside **$CATALINA_HOME** folder.*

> *This file is located in the root folder of the project. However you could also download from here: custom.properties*

*All application properties will be read from this file. If you need to change this path you must modify the class **MainAppConfig***

For more information: https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-external-config.html (section 24.3)

These properties are:

Property	Description
openkm.url	*OpenKM url*
anonymous.user	*Anonymous user name* > *Used to get web services to OpenKM if no user is provided.*
anonymous.password	*Anonymous user password*
admin.user	*Admin user name* > *Used to get web services to OpenKM if no user is provided.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

admin.password	*Admin user password*
preview.download.url	*Document preview url*
logback.file	*Logback file name*

Other properties in this file should not be modified unless you know what are you doing. More information about properties at https://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html

Authentication

This project is configured to authenticate against an OpenKM instance. It works using OpenKM SDK through web services.

OpenKM instance URL is defined in the ***openkm.url*** *configuration property.*

Log file

You must copy file ***logback-custom.xml*** *to* ***$CATALINA_HOME*** *folder. This file is used by log system.*

This file is located in the root folder of the project. However you could download from here: logback-custom.xml.

To change log level for a particular package:

```
<?xml version="1.0" encoding="UTF-8"?>
<included>
  <logger name="com.openkm" level="INFO"/>
</included>
```

Remember me

This sample customization uses H2 database to store remember me data. A database file called ***custom.mv.db*** *will be created into the user home directory.*

There is a file called ***import.sql*** *which contains SQL sentences that are executed when application starts. In this particular case it only contains one sentence to create the database table to allow the "remember me" feature.*

Roles management

In file ***menu.jsp*** *you can see how the roles are managed by the application:*

```
<sec:authorize access="hasRole('CATALOG')">
```

Using this code an user is able to see or not to see this menu option, depending on his roles. However, if you does not want to allow a user to access a resource you can con figure it in ***WebSecurityConfig*** *file.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

*Spring Security 4 automatically prefixes any role with ROLE_*

Bootstrap

The application uses Bootstrap in order to make a beautiful and responsive design. More information about Bootstrap at http://getbootstrap.com/

Packaging

*This application could be packaged as well as a **jar** and **war**. In this pom configuration the artifact will be a war.*

If you need to change packaging please see this url: http://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#build-tool-plugins-maven-packaging (section 66.2)

Application port

If you packaged the application as a jar you could change the port.

To do that please visit: http://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#cli-default-import-statements (section 72.4)

Catalog information

Description

This sample contains a catalog system for OpenKM:

- *Using OpenKM a user could upload files to a folder **cp.pending.folder** (we will se what is this folder)*
- *An automation we will add okg:type metadata to this document*
- *Using the catalog system of this application we will add some metadata to this document*
- *An automation we will move this document to other folder*

The main classes of the catalog system are:

- ***CatalogController**: app routing of the catalog system*
- ***CatalogServiceImpl**: services for the catalog system*
- ***WSCacheDAOImpl**: manages the webservices to communicate OpenKM and this app*

Installation

You must read the following steps in order to configure the application correctly:

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Step 1: Download

Download the application from OpenKM Download center.

> *This will downlaod a zip file contaning a Maven project. For more information about Maven: https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html*

Step 2: Role creation

*Create **ROLE_CATALOG** and add to the user used in the application.*

See Manage roles section for more information.

Step 3: SQL data needed

In order to execute SQL sentences go to Administration > Utilities > Database query and enter these queries:

```
CREATE TABLE CP_DOCUMENT_TYPE(ID INT NOT NULL, TYPE VARCHAR(100) NOT NULL, METADATA_G
CREATE TABLE CP_AGENCY(ID INT NOT NULL, NAME VARCHAR(100) NOT NULL);
INSERT INTO CP_DOCUMENT_TYPE (ID, TYPE, METADATA_GROUP) VALUES (1, 'Consulting', 'okg
INSERT INTO CP_DOCUMENT_TYPE (ID, TYPE, METADATA_GROUP) VALUES (2, 'Marketing', 'okg:
INSERT INTO CP_AGENCY (ID, NAME) VALUES (1, 'Web');
INSERT INTO CP_AGENCY (ID, NAME) VALUES (2, 'Desktop');
```

> *SQL sentences could be different depending on your databse.*

*The parameter **cp.pending.folder** contains the folder to read documents to catalog. You need also to create this folder in the Taxonomy manually:*

```
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('cp.pending.folder', 's
INSERT INTO OKM_CONFIG (CFG_KEY, CFG_TYPE, CFG_VALUE) VALUES ('cp.document.type', 'li
```

*Update the values of the property **cp.document type**:*

```
1;okg:consulting;Consulting
2;okg:marketing;Marketing
```

Step 4: Metadata

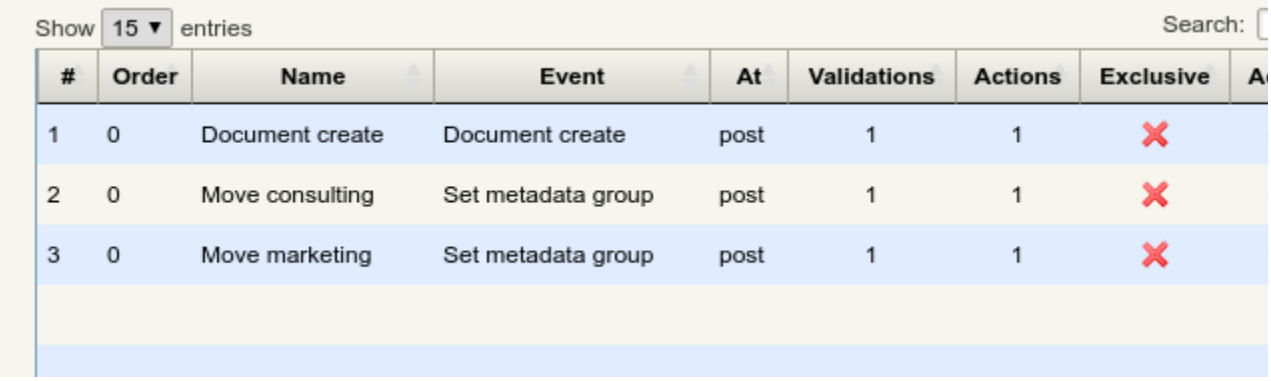
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.9//EN"
                                 "http://www.openkm.com/dtd/property-groups-2.9.dtd">
<property-groups>
  <property-group label="Document type" name="okg:type">
    <select label="Type" name="okp:type.type" type="simple"
            optionsQuery="select ID, TYPE from CP_DOCUMENT_TYPE order by TYPE asc" />
  </property-group>
  <property-group label="Consulting" name="okg:consulting">
    <input label="Date" type="date" name="okp:consulting.date" />
```

```
  </property-group>
  <property-group label="Marketing" name="okg:marketing">
    <select label="Agency" name="okp:marketing.agency" type="simple"
            optionsQuery="select ID, NAME from CP_AGENCY order by NAME asc" />
    <input label="Date" type="date" name="okp:marketing.date" />
  </property-group>
</property-groups>
```

Right now only these tipes are supported: Input, TextArea, Select and Checkbox

Step 5: Automations

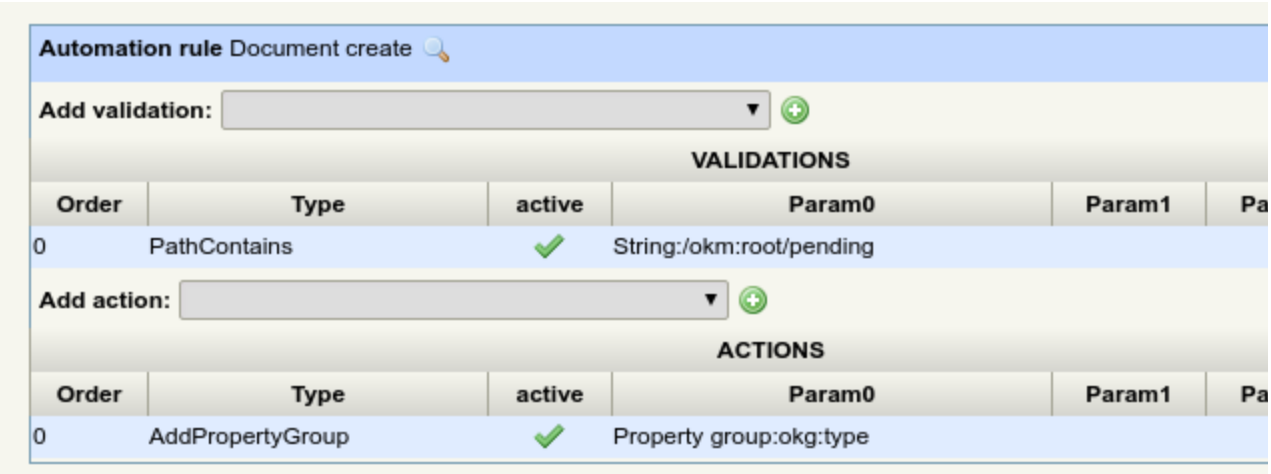
You have to create three automations like the following (see Automation section for more info):



Show 15 entries Search:

#	Order	Name	Event	At	Validations	Actions	Exclusive	A
1	0	Document create	Document create	post	1	1	✖	
2	0	Move consulting	Set metadata group	post	1	1	✖	
3	0	Move marketing	Set metadata group	post	1	1	✖	

1. Add empty type to a document in pending folder



Automation rule Document create

Add validation:

VALIDATIONS

Order	Type	active	Param0	Param1	Pa
0	PathContains	✔	String:/okm:root/pending		

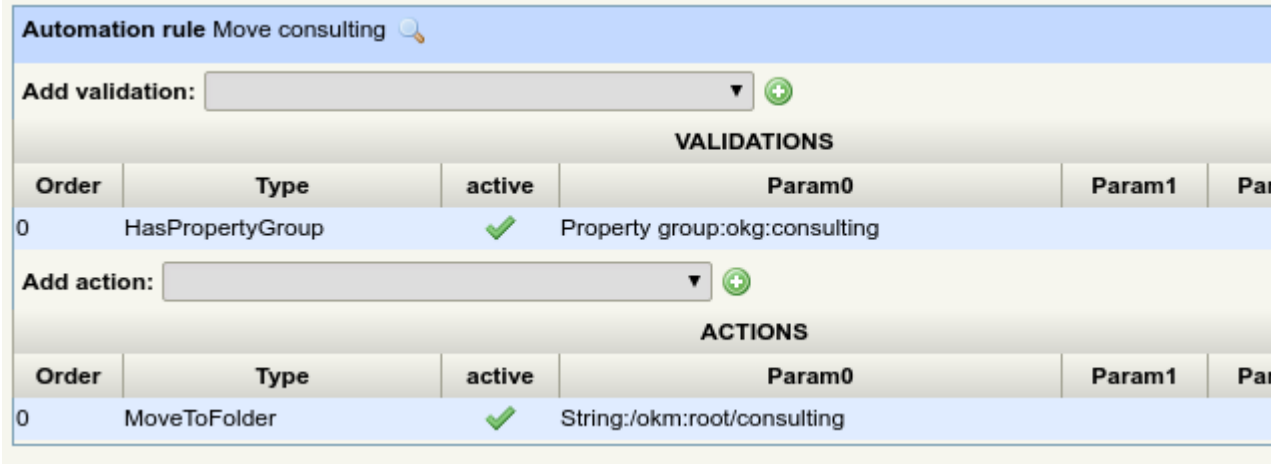
Add action:

ACTIONS

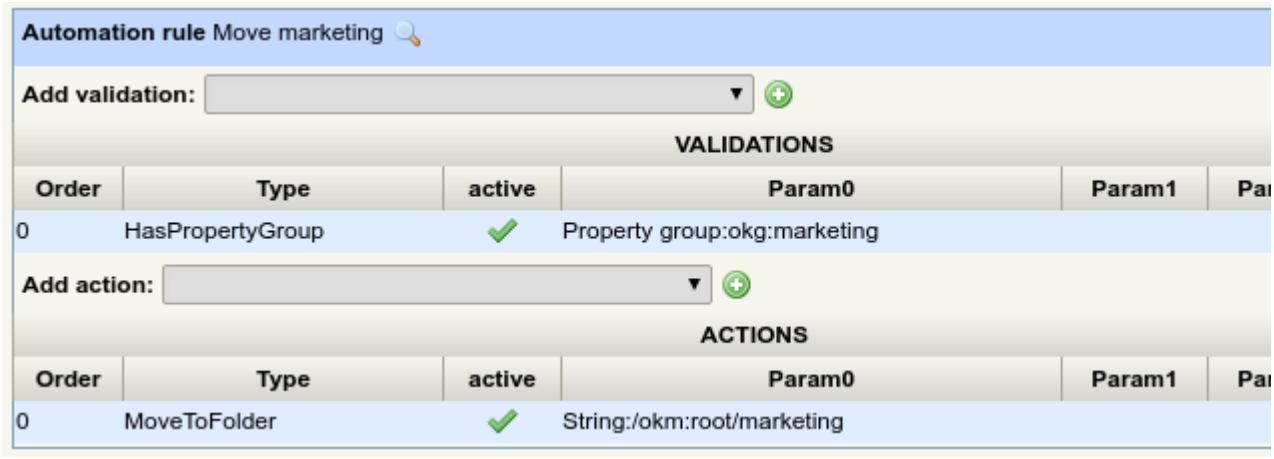
Order	Type	active	Param0	Param1	Pa
0	AddPropertyGroup	✔	Property group:okg:type		

2. Move consulting documents to "consulting" folder

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.



3. Move marketing documents to "marketing" folder



4. Also you have to repeat the steps 2 and 3 but for events "Add metadata group", so you have to create two automations more.

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Form UI template

Developer information for Sample Spring Boot base project with OpenKM authentication and support for JSP, JSTL & REST.

To learn more about Spring Boot, please visit https://projects.spring.io/spring-boot/

There is also a very good tutorial about Spring Boot at https://spring.io/guides/gs/spring-boot/

> *You can download the application (**code-samples.x.x.zip**) from OpenKM Download center.*

Packaging

> *If you are using Eclipse it is mandatory you configure lombok https://projectlombok.org/setup/eclipse*

*This application could be packaged as well as a **jar** and **war**. In this pom configuration the artifact will be a war.*

If you need to change packaging please see this url: http://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#build-tool-plugins-maven-packaging (section 66.2)

Installation

Step 1 - register metadata into OpenKM

- *Go to Administration > Metadata and register the metadata definition:*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 2.1//EN"
                                 "http://www.openkm.com/dtd/property-groups-2.1.dtd">
<property-groups>
  <property-group label="Sample Template" name="okg:tpl">
    <input label="Name" name="okp:tpl.name"/>
    <input label="Birth Date" name="okp:tpl.birth_date" type="date" />
    <select label="Language" name="okp:tpl.language" type="simple">
      <option label="Java" value="java"/>
      <option label="Python" value="python"/>
      <option label="PHP" value="php" />
    </select>
  </property-group>
</property-groups>
```

- *Enable "Sample templates" metadata group in the profiles.*

Step 2 - upload template file

> *You can also download the template.html*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

- *Create file named template.html with content:*

```
<html>
  <body>
    <h1>Sample template</h1>
    <table>
      <tr>
        <td><b>Name</b></td>
        <td>${okp_tpl_name!}</td>
      </tr>
      <tr>
        <td><b>Birth Date</b></td>
        <td>${okp_tpl_birth_date?string("yyyy-MM-dd")}</td>
      </tr>
      <tr>
        <td><b>Language</b></td>
        <td>${okp_tpl_language!}</td>
      </tr>
    </table>
  </body>
</html>
```

- *Upload the **template.html** file to "**/okm:templates/template.html**"*
- *Assign metadata group "Sample template" to the document "**/okm:templates/template.html**"*

Step 3 - deploy the application

*There's a file named **application.properties** into the project where are set the OpenKM connection parameters.*

*By default application should be **run or deployed into the OpenKM server** and exist a user named **okmAdmin** with password **admin** as are set in the properties file shown below:*

```
# Custom properties
openkm.url=http://localhost:8080/OpenKM
openkm.user=okmAdmin
openkm.password=admin
openkm.template=/okm:templates/template.html
```

- *Copy the **forms-X.war** into your **$TOMCAT_HOME**/webapps*

Step 4 - create a document from UI

- *Set in the browser the URL http://localhost:8080/forms-1.0/*
- *Fill the fields shown in the screen and click on submit button*
- *Check that into OpenKM has been created a document*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

template.html

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Security Provider

Developer information for OpenKM authentication thru custom Authentication provider (vía patch) and custom Prinpipal adapter (via plugin).

To learn more about OpenKM authentication providers, please visit Configuring openkm.xml.

To learn more about OpenKM plugins, please visit Register a new plugin.

> *You can download the application (**code-samples.x.x.zip**) from OpenKM Download center.*

General description

Properties file

*You have to modify your called **openkm.properties** inside **$CATALINA_HOME** folder.*

These properties have to be added/modified in this way:

Property	Value
okm.authentication.database	*false*
okm.authentication.supervisor	*false*
okm.authentication.ldap	*false*
okm.authentication.config	*file:/home/openkm/tomcat-8.5.34/openkm.xml* *This is the path where is your tomcat.*

Custom Principal Adapter

You must read the following steps in order to configure this piece correctly:

Step 1: Download

Download the application from OpenKM Download center.

> *This will downlaod a zip file contaning a Maven project. For more information about Maven: https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Step 2: OpenKM configuration

Go to the configuration thru the OpenKM administration ui and search by this property ***principal.adapter*** *and modify its value by* ***com.openkm.plugin.principal.CustomPrincipalAdapter***

Step 3: Install the plugin

Once you build the maven project (security-provider), you have to copy ***security-provider-jar-with-dependencies.jar*** *into the path:* ***$CATALINA_HOME/plugins***

> *You can add/update more users and roles modifying the class* ***com.openkm.security.utils.Users.java***

Authentication Provider

You must read the following steps in order to configure this piece correctly:

Step 1: Download

The application is the same you downloaded from OpenKM Download center.

Step 2: Install the patch

Once you build the maven project (security-provider) copy these two classes:

- *You have to copy the class* ***com.openkm.security.provider.TestCustomAuthProvider.class*** *from* ***security-provider.jar*** *into the path:* ***$CATALINA_HOME/webapps/openkm/WEB-INF/classes/com/openkm/security/provider***
- *And copy* ***com.openkm.security.utils.Users.class*** *from* ***security-provider.jar*** *into the path:* ***$CATALINA_HOME/webapps/openkm/WEB-INF/classes/com/openkm/security/utils***

> *If you added/updated more users and roles, you also have to modify that class* ***com.openkm.security.provider.TestCustomAuthProvider.java***

In addition, you have to configure a patch to preserve that classes across future OpenKM updates. To do this, create a folder called ***$TOMCAT_HOME/patch****. Create an script called* ***prepare.sh*** *with this content:*

```
#!/bin/bash

if [ -f "openkm.war" ]; then
 echo "Patching WAR..."
 jar uf openkm.war WEB-INF/classes/com/openkm/security/provider/TestCustomAuthProvide
 jar uf openkm.war WEB-INF/classes/com/openkm/security/utils/Users.class
 echo "Done!"
else
 echo "Not able to patch file"
fi
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

Remember to check executable permissions of the prepare.sh script.

Step 3: OpenKM configuration

You need this content into your ***openkm.xml***

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns:beans="http://www.springframework.org/schema/beans"
 xmlns:security="http://www.springframework.org/schema/security"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.springframework.org/schema/beans
 http://www.springframework.org/schema/beans/spring-beans.xsd
 http://www.springframework.org/schema/security
 http://www.springframework.org/schema/security/spring-security.xsd">

 <!-- Security delegated to classes -->
 <security:authentication-manager alias="authenticationManager">
 <security:authentication-provider ref="testAuthProvider" />
 </security:authentication-manager>

 <beans:bean id="testAuthProvider" class="com.openkm.security.provider.TestAuthProvid
 </beans:bean>
</beans:beans>
```

Set unique document name based on metadata

Each time a new document is uploaded, the Automation action sets a unique document name based in metadata values.

- *Property **okp:data.id** stores a unique auto incremental value.*
- *Property **okp:data.code** stores a project code value.*
- *Property **okp:data.customer** stores a customer code.*
- *Property **okp:data.description** stores a document description.*
- *When a new document is uploaded the user must fill all the fields except **okp:data.id** which is automatically set by OpenKM.*
- *When Metadata group is changed - event triggered- is executed automation code which generates **okp:data.id** and renames the document based on mask **projectCode-autonumericId-clientCode-description.documentExtension**.*

The tables used in this sample is defined as:

```
-- LastID table
create table UNIQUE_ID (UN_ID INT, primary key(UN_ID));

insert into UNIQUE_ID (UN_ID) values (1);


-- Customer table
create table CUSTOMER (CUS_ID varchar(2),CUS_NAME varchar(32), primary key(CUS_ID));


insert into CUSTOMER (CUS_ID, CUS_NAME) values ('01', 'Customer 01');
insert into CUSTOMER (CUS_ID, CUS_NAME) values ('02', 'Customer 02');
insert into CUSTOMER (CUS_ID, CUS_NAME) values ('03', 'Customer 03');
```

Metadata group definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE property-groups PUBLIC "-//OpenKM//DTD Property Groups 3.8//EN"
"http://www.openkm.com/dtd/property-groups-3.8.dtd">
<property-groups>
    <property-group label="Datos" name="okg:data">
        <input label="Id" type="text" name="okp:data.id" width="200px" readonly="true
        <input label="Project code" type="text" name="okp:data.code" width="200px">
            <validator type="req"/>
            <validator type="num"/>
            <validator type="maxlen" parameter="6"/>
            <validator type="minlen" parameter="6"/>
        </input>
        <select label="Customer" name="okp:data.customer" type="simple"
                optionsQuery="select CUS_ID, CUS_NAME from CUSTOMER order by CUS_NAME
            <validator type="req"/>
        </select>
        <input label="Description" type="text" name="okp:data.description" width="200
```

```
            <validator type="req"/>
            <validator type="maxlen" parameter="150"/>
        </input>
  </property-group>
</property-groups>
```

Automation action:

Take a look at Creating your own Automation Action

```
package com.openkm.plugin.automation.action;

import com.google.gson.Gson;
import com.openkm.core.DatabaseException;
import com.openkm.db.bean.*;
import com.openkm.db.service.LegacySrv;
import com.openkm.module.db.DbDocumentModule;
import com.openkm.plugin.BasePlugin;
import com.openkm.plugin.automation.Action;
import com.openkm.plugin.automation.AutomationException;
import com.openkm.plugin.automation.AutomationUtils;
import com.openkm.util.FileUtils;
import com.openkm.util.PathUtils;
import net.xeoh.plugins.base.annotations.PluginImplementation;
import org.apache.commons.lang3.StringUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
import java.util.stream.Stream;

@PluginImplementation
public class SetUniqueName extends BasePlugin implements Action {
 private static final Logger log = LoggerFactory.getLogger(SetUniqueName.class);
 private static final String PROPERTY_GROUP_NAME = "okg:data";
 private static final String PROPERTY_FIELD_ID_NAME = "okp:data.id";
 private static final String PROPERTY_FIELD_CODE_NAME = "okp:data.code";
 private static final String PROPERTY_FIELD_CUSTOMER_NAME = "okp:data.customer";
 private static final String CODE_PATTERN = "%06d";


 private static final ArrayList<AutomationRule.EnumEvents> EVENTS_AT_PRE = Stream.of(

 private static final ArrayList<AutomationRule.EnumEvents> EVENTS_AT_POST = new Array

 @Autowired
 private AutomationUtils automationUtils;

 @Autowired
 private LegacySrv legacySrv;

 @Autowired
 private DbDocumentModule dbDocumentModule;

 @Override
 public List<AutomationRule.EnumEvents> getValidEventsAtPre() {
  return EVENTS_AT_PRE;
 }
```

```
@Override
public List<AutomationRule.EnumEvents> getValidEventsAtPost() {
 return EVENTS_AT_POST;
}

@Override
public void executePre(Map<String, Object> env, Object... params) throws AutomationE
 // Pre stage, here the metadata still has not been created and it is set the unique
 NodeBase node = automationUtils.getNodeBase(env);
 String propertyGroupName = automationUtils.getPropertyGroupName(env);
 try {
  if (node instanceof NodeDocument && PROPERTY_GROUP_NAME.equals(propertyGroupName))
   // Set the property in the map of values
   Map<String, String> proceedingProperties = automationUtils.getPropertyGroupProper
   proceedingProperties.put(PROPERTY_FIELD_ID_NAME, getUniqueCode());
   // Rename document based in metadata values
   String currentName = PathUtils.getName(node.getPath());
   String currentExtension = FileUtils.getFileExtension(currentName); // Extension m
   // Getting values
   String id = proceedingProperties.get(PROPERTY_FIELD_ID_NAME);
   String code = proceedingProperties.get(PROPERTY_FIELD_CODE_NAME);
   String jsonValueOfCustomer = proceedingProperties.get(PROPERTY_FIELD_CUSTOMER_NAM
   Gson gson = new Gson();
   List<String> customerValues = gson.fromJson(jsonValueOfCustomer, List.class);
   String customerValue = customerValues.get(0);
   // Setting the new name
   String newName = code + "-" + id + "-" + customerValue;
   if (!currentExtension.isEmpty()) {
    newName += "." + currentExtension;
   }
   dbDocumentModule.rename(null, node.getUuid(), newName);
  }
 } catch (Exception e) {
  log.error(e.getMessage(), e);
  throw new AutomationException(e.getMessage(), e);
 }
}

private synchronized String getUniqueCode() throws DatabaseException {
 Integer maxCode = 0;

 StringBuilder query = new StringBuilder("SELECT UN_ID FROM UNIQUE_ID");

 for (List<String> row : legacySrv.executeSQL(query.toString())) {
  maxCode = new Integer(row.get(0));
 }

 // increment value
 legacySrv.executeSQL("UPDATE UNIQUE_ID SET UN_ID = UN_ID +1");

 return String.format(CODE_PATTERN, maxCode);
}

@Override
public void executePost(Map<String, Object> env, Object... params) throws Automation
}

@Override
public String getName() {
 return "Set Unique Name";
}

@Override
public String getParamType00() {
 return Automation.PARAM_TYPE_EMPTY;
}
```

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.

```
  @Override
  public String getParamSrc00() {
   return Automation.PARAM_SOURCE_EMPTY;
  }

  @Override
  public String getParamDesc00() {
   return "";
  }

  @Override
  public String getParamType01() {
   return Automation.PARAM_TYPE_EMPTY;
  }

  @Override
  public String getParamSrc01() {
   return Automation.PARAM_SOURCE_EMPTY;
  }

  @Override
  public String getParamDesc01() {
   return "";
  }

  @Override
  public String getParamType02() {
   return Automation.PARAM_TYPE_EMPTY;
  }

  @Override
  public String getParamSrc02() {
   return Automation.PARAM_SOURCE_EMPTY;
  }

  @Override
  public String getParamDesc02() {
   return "";
  }
 }
```

Create automation rule. Go to Administration > Automation and register a new rule:

- *Rule-based on "**add metadata**" event at the "**pre**" stage.*
- *Set action named "**SetUniqueName**" in the actions.*

Created in 2016 by OpenKM. OpenKM documentation is provided under OpenKM documentation license.