Console command reference

This section lists commands available to the system administrator when connecting to the OSGi console of a running Snow Owl Server instance using telnet or SSH. Instructions for setting up the console can be found in the section titled “OSGi console” of the Configuration guide.

NOTE	Depending on log configuration, the output shown following user-entered commands in the usage examples might only appear in the service log, located at `serviceability/logs/log.log` in the Snow Owl Server installation folder, or the export/import logs in folder `serviceability/logs/snowowl`.

General

To get a quick overview of all available commands, type `help` into the console:

```
osgi> help
---CDO commands---
       cdo list - list all active repositories
       cdo start - start repositories from a config file
       cdo stop - stop a repository
...
service - list all services in the service registry
  scope: vsh
  parameters:
     String   operation (list)

service - examine a service in the service registry
  scope: vsh
  parameters:
     String   operation (examine)
     long     service id

shutdown - shut down the kernel
  scope: vsh
```

The displayed list of commands are a mixture of application-specific and framework-specific operations; the following categories provide functionality related to Snow Owl Server itself:

- Diagnostic and Maintenance commands — `snowowl`
- Terminology Registry commands — `terminologyregistry`
- Session Management commands — `session`
- Scripting commands — `script`
- SNOMED CT Importer commands — `sctimport`
- SNOMED CT OWL Ontology commands — `ontology`
- MRCM commands — `mrcm`

To narrow the command list down to a certain category, type the category name given above:

```
---Snow Owl commands---
    snowowl --version - returns the current version
    snowowl dbcreateindex [nsUri] - creates the CDO_CREATED index on the proper DB
tables for all classes contained by a package identified by its unique namespace URI.
    snowowl listrepositories - prints all the repositories in the system.
    snowowl listbranches [repository] [branchPath] - prints all the child branches of
the specified branch path in the system for a repository. Branch path is MAIN by
default and has to be full path (e.g. MAIN/PROJECT/TASK)
    snowowl reindex [repositoryId] [failedCommitTimestamp] - reindexes the content for
the given repository ID from the given failed commit timestamp (optional, default
timestamp is 1 which means no failed commit).
    snowowl optimize [repositoryId] [maxSegments] - optimizes the underlying index for
the repository to have the supplied maximum number of segments (default number is 1)
    snowowl purge [repositoryId] [branchPath] [ALL|LATEST|HISTORY] - optimizes the
underlying index by deleting unnecessary documents from the given branch using the
given purge strategy (default strategy is LATEST)
    snowowl repositories [repositoryId] - prints all currently available repositories
and their health statuses
```

Disconnecting from the console

To leave the console (and keep the server running), type `disconnect`, then confirm the operation by pressing Enter. Alternatively, you can type "y" to confirm, or "n" to back out.

```
osgi> disconnect
Disconnect from console? (y/n; default=y) y

Connection closed by foreign host.
```

Available commands

Scripting

Execute Groovy script

`script run` parses and executes the specified Groovy script. Services will be provided by the running server instance, similarly to the Groovy editing environment within the Snow Owl client.

Make sure to inspect the server log for any issues, as they might not be printed to the console output depending on the log configuration.

```
osgi> script run /home/user/IndexExample.groovy

[Systemic blood pressure]
Number of results for 'hyp' query term: 3566
[Hypertensive disorder, Hyperlipidaemia, Asthma]
Number of results for 'hyp' AND 'blood cell' query term: 100
Autologous peripheral blood stem cell transplant
White blood cell disorder
...
Red blood cell count, manual, peritoneal fluid
Red blood cell folate borderline high
ID          Label
425983008   Autologous peripheral blood stem cell transplant
54097007    White blood cell disorder
...
442218004   Red blood cell folate borderline high
```

Machine-Readable Concept Model

Import MRCM rules from XMI file

`mrcm import` reads and applies rules from the specified source file. Note that as with regular editing of MRCM rules, only SNOMED CT concept and reference set editors opened after the import will display their output with the changes considered.

This command also requires the specification of a user identifier, which will be presented as the importing user in the commit information view.

```
osgi> mrcm import /path/to/mrcm_20131212143528517.xmi
Impersonate operation as: info@b2international.com

... - Importing MRCM rules...
...
... - Persisting changes...
... - Changes have been successfully persisted.
... - Branch: MAIN Event: SNOMED CT Changes: changed concepts: [123037004:Body
structure], ...
...
... - MRCM rule import successfully finished.
```

Export MRCM rules to an XMI file

`mrcm export` creates a file named `mrcm_{timestamp}.xmi` in the directory given by the administrator. The command requires a user identifier which will be recorded in the user audit log.

```
osgi> mrcm export /path/to/export/folder
Impersonate operation as: info@b2international.com

... - Exporting MRCM rules...
...
... - MRCM rule export successfully finished.
```

Terminology registry

List all imported terminologies

`terminologyregistry listall` displays information about terminologies and terminology extensions imported into the running server instance, including individual local code systems.

```
osgi> terminologyregistry listall

Name: SNOMED CT
Short name: SNOMEDCT
Code System OID: 2.16.840.1.113883.6.96
Maintaining organization link: http://www.snomed.org
Language: ENG
Last version: 0
Current branch path: MAIN
```

SNOMED CT OWL Ontology (reasoner)

Display available reasoners

To list the available reasoners and the preferred one (marked with an * symbol) one has to perform the following:

```
osgi> ontology list

   0 None [version: 4.1.0] (org.protege.editor.owl.NoOpReasoner)
   1 ELK 0.3.2 [version: 0.3.2] (org.semanticweb.elk.elk.reasoner.factory)
 * 2 MORe A (0.1.3) [version: 0.1.3] (org.semanticweb.more.MORe.reasoner.factory)
   3 MORe B (0.1.3) [version: 0.1.3] (org.semanticweb.more.MOReRLrew.reasoner.factory)
   4 FaCT++ [version: 1.6.2] (uk.ac.manchester.cs.owl.factplusplus.factplusplus-
factory)
```

Change the active reasoner

To change the preferred reasoner on the server (in our case from `MORe A` to `FaCT++`), use the following command:

```
osgi> ontology select 4

   0 None [version: 4.1.0] (org.protege.editor.owl.NoOpReasoner)
   1 ELK 0.3.2 [version: 0.3.2] (org.semanticweb.elk.elk.reasoner.factory)
   2 MORe A (0.1.3) [version: 0.1.3] (org.semanticweb.more.MORe.reasoner.factory)
   3 MORe B (0.1.3) [version: 0.1.3] (org.semanticweb.more.MOReRLrew.reasoner.factory)
 * 4 FaCT++ [version: 1.6.2] (uk.ac.manchester.cs.owl.factplusplus.factplusplus-
factory)
```

Note that this setting does not affect ongoing computations if they were started using a different reasoner.

Checking status of available reasoners

The command checks the presence and availability of all reasoners available on the server side. In case of the response below, all reasoners are available.

```
osgi> ontology check
All reasoner instances are available and ready for use.
```

Whenever any of the reasoners is not available, the output should contain the problematic reasoner identifier. Please note that reasoner identifiers may vary; also, if more than one reasoner reports a problem, a list of reasoner identifiers will be printed to the console:

```
osgi> ontology check
Couldn't initialize reasoner factory for ID 'unique.id.of.the.reasoner'.
```

For getting the original cause of the reasoner availability issue, one could dump the exception by appending the `-d` flag:

```
osgi> ontology check -d
Couldn't initialize reasoner factory for ID 'unique.id.of.the.reasoner'.

com.b2international.snowowl.snomed.reasoner.exceptions.ReasonerException: Couldn't
initialize reasoner factory for ID 'unique.id.of.the.reasoner'.
    at
com.b2international.snowowl.snomed.reasoner.server.preferences.ReasonerPreferencesServ
ice.createReasonerInfo(ReasonerPreferencesService.java:306)
    at
com.b2international.snowowl.snomed.reasoner.server.preferences.ReasonerPreferencesServ
ice.checkAllAvailableReasoners(ReasonerPreferencesService.java:270)
    at
com.b2international.snowowl.snomed.reasoner.server.console.SnomedOntologyCommandProvid
er$Command$3.execute(SnomedOntologyCommandProvider.java:65)
    at
com.b2international.snowowl.snomed.reasoner.server.console.SnomedOntologyCommandProvid
er._ontology(SnomedOntologyCommandProvider.java:150)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
    at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
    at java.lang.reflect.Method.invoke(Method.java:597)
    at
org.eclipse.osgi.framework.internal.core.FrameworkCommandInterpreter.execute(Framework
CommandInterpreter.java:209)
...
```

SNOMED CT

Import reference sets in RF2 format from a release text file

Use `sctimport rf2_refset` to import one or more reference sets from an RF2 text file. All RF2 import modes (`FULL`, `SNAPSHOT` and `DELTA`) are available; certain reference set members can be excluded from being imported based on their reference set identifiers.

The following example imports a snapshot release file from the SNOMED CT International RF2 release, excluding two reference sets by identifier:

```
osgi> sctimport rf2_refset /path/to/der2_Refset_SimpleSnapshot_INT_20130731.txt -t
SNAPSHOT -x 447566000 447565001

[2013-12-12 17:10:47.987] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil - SNOMED
CT import started from RF2 release format.
[2013-12-12 17:10:47.987] User: web Event: SNOMED CT import started from RF2 release
format.
Importing release files...: 0% [0ms]
[2013-12-12 17:10:47.988] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Validating release files...
```

```
[2013-12-12 17:10:47.988] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Validating RF2 release files.
[2013-12-12 17:10:47.988] User: web Event: Validating RF2 release files.
[2013-12-12 17:10:47.988] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Creating staging directory '...' for simple type reference set member validation.
Preparing simple type reference set members validation: 5% [96ms]
[2013-12-12 17:10:48.083] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Validating 'der2_Refset_SimpleSnapshot_INT_20130731.txt' release file.
[2013-12-12 17:10:48.083] User: web Event: Validating
'der2_Refset_SimpleSnapshot_INT_20130731.txt' release file.
Validating simple type reference set members...: 11% [868ms]
[2013-12-12 17:10:48.954] [OSGi Console] INFO  c.b.commons.db.JdbcUtils - Connected to
database '...'.
Finishing simple type reference set members validation: 17% [3ms]
[2013-12-12 17:10:48.954] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Preparing simple type reference set member import
[2013-12-12 17:10:48.954] User: web Event: Preparing simple type reference set member
import
[2013-12-12 17:10:48.954] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Creating staging directory '...' for simple type reference set member import.
[2013-12-12 17:10:48.954] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Creating staging directory '...' for simple type reference set member import.
[2013-12-12 17:10:48.954] User: web Event: Creating staging directory '...' for simple
type reference set member import.
Preparing simple type reference set member import: 20% [1ms]
[2013-12-12 17:10:48.955] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Populating storage keys for simple type reference set member import.
[2013-12-12 17:10:48.955] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Populating storage keys for simple type reference set member import.
[2013-12-12 17:10:48.955] User: web Event: Populating storage keys for simple type
reference set member import.
Preparing simple type reference set member import: 21% [2ms]
Preparing simple type reference set member import: 22% [595ms]
Preparing simple type reference set member import: 23% [8382ms]
[2013-12-12 17:10:58.002] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Collecting simple type reference set member import units
[2013-12-12 17:10:58.002] User: web Event: Collecting simple type reference set member
import units
Collecting simple type reference set member import units: 24% [92ms]
...
[2013-12-12 17:10:58.148] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Processing simple type reference set members
[2013-12-12 17:10:58.148] User: web Event: Processing simple type reference set
members
...
Processing simple type reference set members: 88% [17ms]
[2013-12-12 17:10:59.325] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Finishing simple type reference set member import
[2013-12-12 17:10:59.325] User: web Event: Finishing simple type reference set member
import
[2013-12-12 17:10:59.325] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
```

```
Creating indexes for simple type reference set member import.
[2013-12-12 17:10:59.325] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Creating indexes for simple type reference set member import.
[2013-12-12 17:10:59.325] User: web Event: Creating indexes for simple type reference
set member import.
[2013-12-12 17:10:59.334] [OSGi Console] WARN  c.b.s.s.i.rf2.util.ImportUtil -
Couldn't create or drop index SNOMEDREFSET_SNOMEDREFSETMEMBER_IDX1000 for table
SNOMEDREFSET_SNOMEDREFSETMEMBER.
[2013-12-12 17:10:59.334] [OSGi Console] WARN  c.b.s.s.i.rf2.util.ImportUtil -
Couldn't create or drop index SNOMEDREFSET_SNOMEDREFSETMEMBER_IDX1001 for table
SNOMEDREFSET_SNOMEDREFSETMEMBER.
Finishing simple type reference set member import: 89% [10ms]
[2013-12-12 17:10:59.335] [OSGi Console] WARN  c.b.s.s.i.rf2.util.ImportUtil -
Couldn't create or drop index SNOMEDREFSET_SNOMEDREFSETMEMBER_IDX1002 for table
SNOMEDREFSET_SNOMEDREFSETMEMBER.
Finishing simple type reference set member import: 90% [1ms]
[2013-12-12 17:10:59.336] [OSGi Console] WARN  c.b.s.s.i.rf2.util.ImportUtil -
Couldn't create or drop index SNOMEDREFSET_SNOMEDREFSETMEMBER_IDX1003 for table
SNOMEDREFSET_SNOMEDREFSETMEMBER.
Finishing simple type reference set member import: 91% [1ms]
[2013-12-12 17:10:59.336] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil -
Removing staging directory '...' from simple type reference set member import.
[2013-12-12 17:10:59.336] [OSGi Console] INFO  c.b.s.s.i.r.m.AbstractSnomedImporter -
Removing staging directory '...' from simple type reference set member import.
[2013-12-12 17:10:59.336] User: web Event: Removing staging directory '...' from
simple type reference set member import.
Finishing simple type reference set member import: 94% [1ms]
Finishing simple type reference set member import: 100% [0ms]
[2013-12-12 17:10:59.337] [OSGi Console] INFO  c.b.s.s.i.rf2.util.ImportUtil - SNOMED
CT import successfully finished.
[2013-12-12 17:10:59.337] User: web Event: SNOMED CT import successfully finished.
```

Note that messages related to not being able to create or drop database indexes are not an indication of a failed import process.

Import reference sets in DSV format

`sctimport dsv_refset` imports a single reference set from text files in Delimiter Separated Values (DSV) format. The syntax is as follows:

```
sctimport dsv_refset <path> <hasHeader> <skipEmptyLines> <parentConcept>
```

<path>

Specifies the file to be used for importing

<hasHeader>

Set to `true` if the source text file has a header row, `false` otherwise

<skipEmptyLines>

Set to `true` if the source text file has empty lines which should be ignored, `false` otherwise

<parentConcept>

Set to an integer value specifying the parent of the imported reference set's identifier concept

Accepted values for parentConcept are:

1. Simple type
2. B2i examples
3. KP Convergent Medical Terminology
4. CORE Problem List
5. Infoway Primary Health Care

The reference set name is determined by the input file name; as an example, `CamelCase.csv` will be converted to `Camel Case reference set`. An attempt will be made to interpret the first column of each line as a SNOMED CT concept identifier. If the identifier can be resolved, a member will be added to the reference set, otherwise an exception is thrown.

```
osgi> sctimport dsv_refset /path/to/SampleConcepts.txt false true 0
Impersonate operation as: info@b2international.com

Importing Interesting Reference Set...
[2013-12-12 17:22:03.154] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionBeforeCommitting() start
[2013-12-12 17:22:03.154] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionBeforeCommitting() lock acquired for BranchPath{Path='MAIN'}
[2013-12-12 17:22:03.155] [Worker-36] INFO
c.b.s.s.r.s.c.SnomedReasonerChangeProcessor - >>> Processing OWL ontology changes
[2013-12-12 17:22:03.155] [Worker-36] INFO
c.b.s.s.r.s.c.SnomedReasonerChangeProcessor - --- Processing OWL ontology changes:
change processing skipped, no ontology instance present for branch or running in
embedded mode
[2013-12-12 17:22:03.156] [Worker-36] INFO
c.b.s.s.r.s.c.SnomedReasonerChangeProcessor - <<< Processing OWL ontology changes
[249.6 µs]
[2013-12-12 17:22:03.156] [Worker-43] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Processing and updating changes...
[2013-12-12 17:22:03.156] [Worker-43] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Retrieving taxonomic information from store.
[2013-12-12 17:22:03.313] [Worker-42] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Processing changes taxonomic information.
[2013-12-12 17:22:03.313] [Worker-34] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Building taxonomic information.
[2013-12-12 17:22:03.315] [Worker-42] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Rebuilding taxonomic information based on the changes.
[2013-12-12 17:22:03.573] [Taxonomy difference processor] INFO
```

```
c.b.s.d.s.s.i.SnomedCDOChangeProcessor - Calculating taxonomic differences...
[2013-12-12 17:22:03.586] [Worker-43] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Updating reference set membership changes...
[2013-12-12 17:22:03.586] [Worker-43] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Updating taxonomy...
[2013-12-12 17:22:03.602] [Taxonomy difference processor] INFO
c.b.s.d.s.s.i.SnomedCDOChangeProcessor - Calculating taxonomic differences
successfully finished.
[2013-12-12 17:22:03.602] [Worker-43] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Processing and updating changes successfully finished.
[2013-12-12 17:22:03.602] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionBeforeCommitting() end
[2013-12-12 17:22:03.628] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionAfterCommitted() start
[2013-12-12 17:22:03.628] [Worker-36] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Persisting changes...
[2013-12-12 17:22:03.689] [Worker-36] INFO  c.b.s.d.s.s.i.SnomedCDOChangeProcessor -
Changes have been successfully persisted.
[2013-12-12 17:22:03.689] User: info@b2international.com Branch: MAIN Event: SNOMED CT
Changes: new concepts added: [745288891000154109:Sample Concepts reference set],
changed concepts: [446609009:Simple type reference set], new reference sets:
[745288891000154109:Sample Concepts reference set],
[2013-12-12 17:22:03.690] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionAfterCommitted() end
[2013-12-12 17:22:03.690] [OSGi Console] INFO  c.b.s.d.s.CDOServerChangeManager -
handleTransactionAfterCommitted() lock released for BranchPath{Path='MAIN'}
[2013-12-12 17:22:03.769] [OSGi Console] INFO  c.b.s.d.PostStoreUpdateManager - Commit
notification received for user info@b2international.com.
All concepts were imported.
```

Diagnostics and maintenance

`snowowl --version` prints the current server version.

```
osgi> snowowl --version
 5.8.0
```

`snowowl repositories [id]` prints information about all repositories or just a single single repository

```
osgi> snowowl repositories
 ---------------------------------------------------
 |id              |health          |diagnosis       |
 ---------------------------------------------------
 |snomedStore     |GREEN           |-               |
 ---------------------------------------------------
```

`snowowl listbranches [repository]` prints all the branches in the system for a repository.

```
osgi> snowowl listbranches snomedStore
Branches for repository snomedStore:
|---MAIN
    |---2011-10-01
    |---2012-01-31
    |---2012-07-31
    |---2013-01-31
    |---2013-07-31
    |---2014-01-31
    |---2014-07-31
    |---2015-01-31
        |---SNOMED-CT-SE
            |---2012-12-21
            |---2013-05-31
            |---2013-11-30
            |---2014-05-31
            |---2014-11-30
            |---2015-05-31
    |---2015-07-31
    |---2016-01-31
        |---SNOMED-CT-SE
```

`snowowl dbcreateindex [nsUri]` creates the CDO_CREATED index on the proper DB tables for all classes contained by a package identified by its unique namspace URI.

`snowowl reindex [repositoryId]` recreates the entire index for the content of the given repository. This long-running process requires the the server to be shut-down, the index to be deleted manually and a restart before invoking this command.

```
osgi>snowowl reindex snomedStore
```

`snowowl optimize [repositoryId] [maxSegments]` optimizes the underlying index for the repository to have the supplied maximum number of segments (default number is 1).

```
osgi>snowowl optimize snomedStore 6
```

`snowowl purge <repositoryId> <branchPath> <purgeStrategy>` optimizes the underlying index by deleting unnecessary documents from the given branch using the given purge strategy (default strategy is LATEST, available strategies are ALL, LATEST, HISTORY)

```
osgi>snowowl purge snomedStore MAIN/2016-01-31 ALL
```

Session management

Display connected users and session identifiers

To list all connected users (and the unique session ID), run the following OSGi command:

```
osgi> session users

User: akitta@b2international.com | session ID: 3
User: obali@b2international.com | session ID: 4
User: zstorok@b2international.com | session ID: 5
User: apeteri@b2international.com | session ID: 6
```

Send message to users

To send message to all connected users, use the following command:

```
osgi> session message ALL Some message from the administrator.

Message sent to akitta@b2international.com
Message sent to obali@b2international.com
Message sent to zstorok@b2international.com
Message sent to apeteri@b2international.com
```

All connected client will receive the message via a dialog.

For sending message to a subset of recipient users, execute the following OSGi command:

```
osgi> session message obali@b2international.com,zstorok@b2international.com Message
from the administrator to Orsi and Zsolt.

Message sent to obali@b2international.com
Message sent to zstorok@b2international.com
```

Restrict user logins

Administrator may restrict non-administrator user log in to the sever with the following:

```
osgi> session login disabled

Disabled non-administrative logins.
```

Users will not be able to connect to the server while non-administrator log in is disabled. Clients will receive the following when trying to connect to the server from the splash screen:

```
Logging in for non-administrator users is temporarily disabled.
```

Invoking this command will not disconnect any of the connected non-administrator users. The way how to disconnect clients from the server will be discussed below.

To re-enable non-administrator log in onto the server refer to the following command:

```
osgi> session login enabled

Enabled non-administrative logins.
```

The status can be checked with the following command:

```
osgi> session login status

Non-administrative logins are currently enabled.
```

Disconnect users

To disconnect a subset of connected users from the server, the following command should be performed:

```
osgi> session disconnect akitta@b2international.com,apeteri@b2international.com

User: akitta@b2international.com ,session id: 3 was disconnected.
User: apeteri@b2international.com ,session id: 6 was disconnected.
```

All disconnected users will receive a message about the lost connection. Then client application could be closed gracefully. This will not prevent users to reconnect the server.

The recommended way to ensure that none of the users are connected to the server when performing any single system administrator task is the following:

- Disable non-administrator log in in the server
- Notify users about the upcoming system admin operation
- Disconnect all users

Repository management

Display terminology repositories

To list all available repositories and their identifiers, one should execute the following command:

```
osgi> session repositories

   LOINC Store [ID: loincStore]
   Local Code System Store [ID: localterminologyStore]
   Terminology Metadata Store [ID: terminologyregistryStore]
   ICD-10 Store [ID: icd10Store]
   Value Set Store [ID: valuesetStore]
   ATC Store [ID: atcStore]
   SNOMED CT Store [ID: snomedStore]
   Mapping Set Store [ID: mappingsetStore]
   ICD-10-AM Store [ID: icd10amStore]
```

Display currently held locks

To view a table of acquired locks, their targets and owners, execute the command:

```
osgi> session showlocks
No locks are currently granted on this server.

osgi> session lock allrepositories
Acquired lock for all repositories.

osgi> session showlocks

  Id | Lvl | Created on       | Locked area       | Owner context
-------------------------------------------------------------------------------------
----------------
   0 |   1 | 2014-01-31 21:16 | All repositories  | Lock owner: System
     |     |                  |                   | Performing maintenance from the
server console
-------------------------------------------------------------------------------------
----------------

 osgi> session lock allrepositories
 Acquired lock for all repositories.

  Id | Lvl | Created on       | Locked area       | Owner context
-------------------------------------------------------------------------------------
----------------
   0 |   2 | 2014-01-31 21:16 | All repositories  | Lock owner: System
     |     |                  |                   | Performing maintenance from the
server console
-------------------------------------------------------------------------------------
----------------
```

Locks can be acquired for different purposes, such as:

- administrative maintenance

- backup
- saving editors
- classification

Their area of effect can also vary:

- all terminology stores
- a single terminology store
- a single branch of a particular terminology store

Once a lock owner obtains a lock, the associated area is available for their use only; others will receive indications that someone else is already working on something which requires uninterrupted access to the target area. Lock attempts on the same or overlapping areas will not be able to complete until the lock is released.

The "Lvl" column indicates the "nesting" count of a granted lock; when someone holds a lock, they can lock the same area multiple times. Ownership is only released when the level decreases to 0.

Lock and release areas of terminology stores

To lock all terminology stores simultaneously, issue the following command:

```
osgi> session lock allrepositories
Acquired lock for all repositories.
```

If a conflicting lock has already been acquired by a different owner, the reason for not granting this request will be displayed in the response.

To lock all branches of a single terminology store, refer to the repository identifiers returned by the `session repositories` command:

```
osgi> session lock snomedStore
Acquired lock for repository 'snomedStore'.
```

Similarly, for locking a single branch of a single repository, type:

```
osgi> session lock snomedStore MAIN/a
Acquired lock for branch 'MAIN/a' of repository 'snomedStore'.

osgi> session showlocks

  Id | Lvl | Created on       | Locked area                                | Owner
context
-------------------------------------------------------------------------------------
-------------------------------------
   0 |   1 | 2014-01-31 21:16 | All repositories                           | Lock owner:
System
     |     |                  |                                            | Performing
maintenance from the server console
-------------------------------------------------------------------------------------
-------------------------------------
   1 |   1 | 2014-01-31 21:30 | Repository 'snomedStore'                   | Lock owner:
System
     |     |                  |                                            | Performing
maintenance from the server console
-------------------------------------------------------------------------------------
-------------------------------------
   2 |   1 | 2014-01-31 21:30 | Branch 'MAIN' of repository 'snomedStore'| Lock owner:
System
     |     |                  |                                            | Performing
maintenance from the server console
-------------------------------------------------------------------------------------
-------------------------------------
```

The branch path argument is case sensitive; as an example, `main`, `Main`, `main/a` and `Main/a` branch paths would be invalid arguments.

Releasing an owned lock can be performed by executing a corresponding `session unlock` command:

```
osgi> session unlock snomedStore MAIN
Released lock for branch 'MAIN' of repository 'snomedStore'.

osgi> session showlocks

  Id | Lvl | Created on       | Locked area             | Owner context
-------------------------------------------------------------------------------
----------------------
   0 |   1 | 2014-01-31 21:16 | All repositories        | Lock owner: System
     |     |                  |                         | Performing maintenance from
the server console
-------------------------------------------------------------------------------
----------------------
   1 |   1 | 2014-01-31 21:30 | Repository 'snomedStore'| Lock owner: System
     |     |                  |                         | Performing maintenance from
the server console
-------------------------------------------------------------------------------
----------------------
```

NOTE	Regular save operations try to get the lock for their target repository and branch. If the administrator has already taken over an area by using the commands above, a dialog will be displayed when the user tries to save after approx. 5 seconds of waiting for the lock to be granted.

Forcefully unlock stuck locks

If an operation gets stuck, or otherwise fails to release locks for which the System user is not the owner, other users may be blocked indefinitely as a result. To resolve the situation, one can forcefully unlock such locks by referring to their identifier shown in the table:

```
osgi> session forceunlock 1
Forcefully released lock with identifier 1.

 osgi> session showlocks

  Id | Lvl | Created on       | Locked area             | Owner context
-------------------------------------------------------------------------------
----------------------
   0 |   1 | 2014-01-31 21:16 | All repositories        | Lock owner: System
     |     |                  |                         | Performing maintenance from
the server console
-------------------------------------------------------------------------------
----------------------
```

To forcefully release all locks, use the command with the `all` argument instead of the identifier.

Remote jobs

Remote jobs are long-running operations intended to be executed on the server; the requesting user's client need not be kept open while the job is active. The result of these computations can be checked immediately, or at a later time, if the results are still available for review. The administration console provides two commands for listing currently running remote jobs, and requesting cancellation of these items.

Display remote jobs

To list all currently scheduled, running, or finished remote jobs, type:

```
osgi> remotejobs list

  Id | Description                    | Owner            | Scheduled        |
Started          | Status
-------------------------------------------------------------------------------
----------------------
   0 | Batch ontology generation      | info@b2intern... | 2014-03-20 11:03 | 2014-
03-20 11:03 | Finished
-------------------------------------------------------------------------------
----------------------
   1 | Classifying the ontology on MAIN | info@b2intern... | 2014-03-20 11:13 | 2014-
03-20 11:13 | Finished
-------------------------------------------------------------------------------
----------------------
```

(Note that identifiers in the first column can change at any time if, for example, the initiator of the task removes a completed job from their list.)

Cancel remote job

To signal a remote job that it should finish its work at the closest possible occasion without completing it fully, use the following command with the identifier from the list displayed above. If `remotejobs list` was not invoked earlier, the following message will be printed to the console:

```
osgi> remotejobs cancel 0
Please retrieve the list of currently scheduled or running jobs first.
```

Otherwise, when a valid job identifier is given, the following output should appear:

```
osgi> remotejobs cancel 0
Requesting job 0 to cancel.
```

...and an additional invocation of `remotejobs list` should list the given job's status as "Cancel requested".

NOTE Not all remote jobs are able to react to cancel requests.