Administration

This section covers administrative and maintenance tasks in Snow Owl Server.

Startup and shutdown

Scripts for starting and stopping a Snow Owl Server instance are located in `/opt/snowowl-{edition}_{version}/bin/*.sh` files. Change the active user to `snowowl` when starting the server:

```
# su snowowl -s /bin/bash -c "nohup bin/startup.sh > /dev/null" &
[1] 12473
nohup: ignoring input and redirecting stderr to stdout

# jobs
[1]+  Running    su snowowl -s /bin/bash -c "nohup bin/startup.sh > /dev/null" &
```

Note that startup will continue in the background and the server stays running even if the user disconnects from the server.

You can follow the startup sequence by inspecting `serviceability/logs/log.log`:

```
# tail -f serviceability/logs/log.log
...
[2017-01-24 02:50:12.753] INFO  start-signalling-2 WE0001I Started web bundle [...]
[2017-01-24 02:50:12.756] INFO  start-signalling-2 DE0005I Started bundle [...]
[2017-01-24 02:50:12.756] INFO  start-signalling-2 Thread context class loader [...]
[2017-01-24 02:50:12.757] INFO  start-signalling-2 DE0005I Started plan
'osgi_server.plan' version '1.0.0'.
```

Snow Owl Server finishes the startup procedure after displaying the lines from above. You can verify that everything is working properly by checking the server’s version, the available databases and their health states via the OSGi console:

```
# yum install telnet ①

# telnet localhost 2501
Trying ::1...
Connected to localhost.
Escape character is '^]'.

osgi> snowowl --version ②
5.8.6

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

osgi> disconnect
Disconnect from console? (y/n; default=y) y
Connection closed by foreign host.
```

① Install the client first (if not already present)

② Should always print the currently running server's version

The server can be stopped cleanly by running `bin/shutdown.sh`:

```
# cd /opt/snowowl-{edition}_{version}
# bin/shutdown.sh
```

Importing SNOMED CT content

To import a `SnomedCT_Release_XXX_xxxxxxxx.zip` release archive, the file must be present in the Snow Owl Server host file system, which we will refer to as {import_archive}.

Starting the import

Use the following OSGi command to start the import process:

```
osgi> sctimport rf2_release -t <type> -b <branch> -v /path/to/{import_archive}
/path/to/{release_descriptor_file}
```

-t <type>

One of `FULL`, `SNAPSHOT` or `DELTA`, depending on the import type the administrator wants to use. The parameter is case-insensitive.

-b <branch>

The existing branch to import the content onto. In case of extension import, an effective time from the base SNOMED CT release (e.g. 2016-01-31). If omitted `MAIN` will be used.

-v

Creates versions for each effective time found in the release archive. If omitted no versions will be created.

/path/to/{import_archive}

Specifies the release archive to import. Must be a `.zip` file with a supported internal structure, such as the release archive of the International Release.

/path/to/{release_descriptor_file}

The path to the release descriptor `.json` file.

Release decriptor file

The release descriptor `.json` file could contain the following attributes as key/value pairs:

Name

Descriptive name of the code system.

Short name

Short name of the code system. Must **not** contain any whitespaces, must be **unique** among the other code systems already present in Snow Owl.

TIP See `terminologyregistry listall` OSGi command to list all existing code systems.

Language

Three letter language code of the code system.

Code system OID

The OID of the code system.

Extension of

The short name of the base code system / release this SNOMED CT release depends on.

Maintaining organization link

Maintaining organization link.

Release type

The type of the release. Either `INTERNATIONAL` or `EXTENSION`.

Citation

Citation.

Examples

International import

```
osgi> sctimport rf2_release -t FULL /path/to/{international_import_archive}
/path/to/snomed_ct_international.json
```

Where `snomed_ct_international.json` looks like:

```
{
  "name": "Systematized Nomenclature of Medicine Clinical Terms International Version
",
  "shortName": "SNOMEDCT",
  "language": "ENG",
  "codeSystemOID": "2.16.840.1.113883.6.96",
  "maintainingOrganizationLink": "http://www.ihtsdo.org",
  "citation": "SNOMED CT contributes to the improvement of patient care by
underpinning the development of Electronic Health Records that record clinical
information in ways that enable meaning-based retrieval. This provides effective
access to information required for decision support and consistent reporting and
analysis. Patients benefit from the use of SNOMED CT because it improves the recording
of EHR information and facilitates better communication, leading to improvements in
the quality of care."
}
```

Extension import

An extension import based on the 2015-01-31 version of the international release:

```
osgi> sctimport rf2_release -t FULL -b 2015-01-31 -v
/path/to/{extension_import_archive} /path/to/snomed_ct_extension.json
```

Where `snomed_ct_extension.json` looks like:

```
{
  "name": "SNOMED CT Special Extension",
  "shortName": "SNOMEDCT-SE",
  "language": "ENG",
  "codeSystemOID": "2.16.840.1.113883.6.96.2",
  "extensionOf": "SNOMEDCT",
  "maintainingOrganizationLink": "http://www.snomed-special-extension.org",
  "citation": "Long citation about the details of SNOMED CT special extension"
}
```

NOTE While the import is running, feedback might be delayed on the OSGi console. Log output can be observed throughout the import session in the file `serviceability/logs/log.log`. The import may take several hours depending on your hardware and JVM configuration.

Single administrator operations

Import and export processes are dedicated single administrator operations. As these operations are long-running, administrators need to ensure that during these processes no other users should be connected to the system. The following steps describe how to disconnect all clients from the server and how to ensure that no one else, but the administrator is the only connected user while performing any import/export operations.

Steps to perform single admin operations

Checking the connected users from the OSGi server console, to list all connected users one should perform the following command:

```
osgi> session users
User: info@b2international.com ,session id: 9
```

Before starting to gracefully disconnect users, the administrator should disable non-administrator user logins to the server. To check the login status on the server:

```
osgi> session login status
Non-administrative logins are currently enabled.
```

As the response states above, there is no login restrictions applied. To restrict non-administrator logging, one should execute the following command:

```
osgi> session login disabled
Disabled non-administrative logins.
```

Now any users with insufficient privileges (in other words; users without 'Administrator' role) will be refused by the server when trying to connect.

NOTE None of the currently connected users will be disconnected. Connected users have to be disconnected by the administrator via the OSGi console as described later.

The administrator can send an informational message from the OSGi console to connected clients, so users can be informed about the upcoming maintenance:

```
osgi> session message ALL Server is going down in 10 minutes due to a SNOMED CT
publication process. Please commit all your unsaved changes.
Message sent to info@b2international.com
```

To disconnect all currently connected users:

```
osgi> session disconnect ALL
User: info@b2international.com ,session id: 9 was disconnected.
```

NOTE In this case, all clients, including the administrator will be logged out from the server, but the administrator may reconnect to the server as only non-administrative users are locked out.

After disabling non-administrator user login, notifying and disconnecting users, double-check of the current status and the connected users at the server:

```
osgi> session login status
Non-administrative logins are currently disabled.
```

```
osgi> session users
osgi>
```

It is now safe to perform any single administrator operations, such as an RF2 import. When finished, enable non-administrative connections again:

```
osgi> session login enabled
Enabled non-administrative logins.
```

Impersonating users

Snow Owl Server will ask for a user identifier for server-side import operations in the following cases:

- SNOMED CT RF2 import
- SNOMED CT MRCM import

The user identifier will be used for associating commits to the terminology repository with a user in the commit information view.

Taking backups

"Hot" backups

The example shell script `snowowl_hot_backup_mysql.sh` exercises all functionality mentioned above, and produces a .zip archive containing database dumps and copies of index folders in the directory it is started from. Please update the variable `SNOW_OWL_SERVER_HOME` so that it points to the installation folder of Snow Owl Server before running the script.

The return value is 0 for successful backups, and 1 if an error occurs while backing up content from the server. The script produces timestamped diagnostic output on its standard output; error messages are directed to the standard error output.

To create backups regularly, add a dedicated non-login user for backups as root:

```
# useradd -r -M -d / -s /sbin/nologin -c "Snow Owl Backup" snowowl-backup
```

Create and/or update access privileges of the backup destination, log output, and the location of the singleton instance lock file:

```
# mkdir -pv /storage/backups /var/log/snowowl-backup /var/run/snowowl-backup
mkdir: created directory `/storage/backups'
mkdir: created directory `/var/log/snowowl-backup'
mkdir: created directory `/var/run/snowowl-backup'

# chown -v root:snowowl-backup /storage/backups /var/log/snowowl-backup
/var/run/snowowl-backup
changed ownership of `/storage/backups' to root:snowowl-backup
changed ownership of `/var/log/snowowl-backup' to root:snowowl-backup
changed ownership of `/var/run/snowowl-backup' to root:snowowl-backup

# chmod -v 775 /storage/backups /var/log/snowowl-backup /var/run/snowowl-backup
mode of `/storage/backups' changed to 0775 (rwxrwxr-x)
mode of `/var/log/snowowl-backup' changed to 0775 (rwxrwxr-x)
mode of `/var/run/snowowl-backup' changed to 0775 (rwxrwxr-x)
```

Save the backup script in an accessible place, set the owner to snowowl-backup, and make it executable:

```
# chown -v snowowl-backup: /storage/backups/snowowl_full_backup_mysql.sh
changed ownership of `/storage/backups/snowowl_full_backup_mysql.sh' to snowowl-
backup:snowowl-backup

# chmod -v 744 /storage/backups/snowowl_full_backup_mysql.sh
mode of `/storage/backups/snowowl_full_backup_mysql.sh' changed to 0744 (rwxr--r--)
```

Add the script to the backup user’s crontab (the example runs the script at 4 AM, and outputs log entries to logfiles with a year-month-date suffix in /var/log/snowowl-backup):

```
# EDITOR=nano crontab -e -u snowowl-backups

<nano opens; add the content below to the opened file, save, and exit the editor>

# MAILTO="local-user"
#
# Minute - Hour - Day of month - Month - Day of week - Command
0 4 * * * cd /storage/backups && ( ./snowowl_full_backup_mysql.sh >> /var/log/snowowl-
backup/log-`date +\%Y\%m\%d` 2>&1 )
```

(If the standard error output is not redirected with the "2>&1" part of the command, errors will be captured by cron and mailed to the snowowl-backup user's mailbox. The destination can be changed by uncommenting the MAILTO parameter and setting it to a different address.)

"Cold" backups

When the server is shut down, the above mentioned REST service for enumerating store content and getting exclusive write locks for the repositories is not available, so a separate script, `snowowl_cold_backup_mysql.sh` is being provided for this case.