Softneta

UAB Softneta
K. Barsausko st. 59, LT 51423
Kaunas, Lithuania
info@softneta.com
www.softneta.com

MedDream DICOM Viewer

Install MANUAL

(version 7.5.2, 2020-09-11)

© 2020, Softneta UAB, Kaunas

All rights reserved in the event of granting of patents or registration as a utility patent.

All names of companies and products mentioned in this manual may be trademarks or registered trademarks. References to products of other manufacturers are for information purposes only. Such references are intended neither as an approval nor a recommendation of these products. Softneta UAB accepts no liability for the performance or use of such products.

Other brand names, software and hardware names used in this manual are subject to trademark or patent protection. The quoting of products is for informational purposes only and does not represent a trademark misuse.

This Install Manual is protected by copyright. Unless expressly authorized in writing, dissemination, duplication or other commercial exploitation of this documentation set or communication of its contents or parts of it is not permitted. In case of infringement, the violator may be liable to pay compensation for damages.

Specifications due to technical developments are subject to change. This Install Manual is not subject to the revision service. Please contact the manufacturer or authorized dealer to request the latest edition of Install Manual.

Table of Contents

1 Introduction **1**

2 Minimal server side requirements **1**
2.1 Minimal hardware requirements . . . 1
2.2 Minimal memory requirements . . . 2
2.3 Minimal software requirements . . . 2
2.3.1 Supported operating systems . . . 2
2.3.2 Supported PHP scripting interpreters . . . 2
2.3.3 Supported Java runtime . . . 2
2.3.4 Decommissioning of MedDream . . . 3

3 Integration with PACS **3**
3.1 Integration modes . . . 3
3.1.1 Direct access mode on a different host . . . 3
3.1.2 Saving measurements and Key Objects . . . 4
3.1.3 Multiple PACSes in parallel . . . 4
3.2 Direct access to PACS database mode . . . 4
3.2.1 PacsOne . . . 4
3.2.2 DCM4CHEE . . . 9
3.2.3 ClearCanvas . . . 13
3.2.4 Conquest . . . 14
3.3 Integration over Web or other APIs . . . 16
3.3.1 Orthanc . . . 16
3.3.2 DICOMweb . . . 17
3.4 DICOM mode . . . 20
3.4.1 DICOM mode notes . . . 20
3.4.2 MedDream configuration for DICOM mode . . . 21
3.5 File system access mode . . . 22
3.5.1 File system mode notes . . . 22
3.5.2 MedDream configuration for File system mode . . . 23

4 Image Access from hospital information system (HIS) **24**

5 Deployment **25**
5.1 Configuration . . . 25
5.1.1 Essential configuration and the first run . . . 25
5.1.2 System.json . . . 27
5.1.3 Reference for remaining configuration parameters . . . 29
5.1.4 Built-in authentication and authorization . . . 37
5.1.5 Predefined login credentials . . . 42
5.1.6 Reporting . . . 42
5.1.7 Java memory limits . . . 43
5.1.8 Enabling SSL . . . 44
5.1.9 The external links plugin . . . 45
5.2 Running as a service . . . 47
5.2.1 Windows . . . 47
5.2.2 Linux (System-V init) . . . 47
5.2.3 Linux (systemd) . . . 48
5.2.4 Linux (upstart) . . . 49
5.3 Verification checklist . . . 49
5.4 Option: dockerized environment . . . 51

6 Additional software **51**
6.1 3D service . . . 51
6.1.1 3D server configuration . . . 51

6.1.2 Network settings (ports and SSL) . . . 51
6.1.3 Quality / Performance . . . 51
6.2 Browser plugin . . . 53

7 Security considerations **53**
7.1 Default passwords . . . 53
7.2 Search engines . . . 53
7.3 DCM4CHEE 2.x . . . 53
7.4 Embedding into IFRAME etc . . . 54
7.5 php.ini . . . 54
7.6 Browser's cache . . . 57
7.7 Firewalls . . . 58
7.8 Summary: Minimum relevant IT security requirements . . . 58
7.9 Summary: Minimum relevant IT security requirements . . . 59

8 Localization **60**

9 Rebranding **60**

10 Troubleshooting **63**
10.1 Log files of the Java-based core . . . 63
10.2 Browser download path selection . . . 64
10.2.1 Google Chrome . . . 64
10.2.2 Mozilla Firefox . . . 64
10.2.3 Microsoft Edge . . . 64
10.2.4 Safari . . . 65
10.2.5 Internet Explorer 11 . . . 65
10.3 Symptom: on Linux the loading pauses just after logging in, possibly with wrong webpage colors . . . 65

A Appendix: Configuration changes since 7.5.1 **66**
A.1 application.properties, generic . . . 66
A.1.1 Renamed options . . . 66
A.1.2 Options that are handled differently . . . 66
A.1.3 Options with different visibility . . . 67
A.1.4 New options or values . . . 67
A.2 application.properties, Conquest plugin 1.1.1 . . . 68
A.2.1 Renamed options . . . 68
A.3 (since 7.1.1) application.properties, FileSystem plugin 2.1.0 . . . 68
A.3.1 Options that are handled differently . . . 68
A.3.2 Options with different visibility . . . 68
A.4 system.json . . . 68
A.4.1 New options . . . 68
A.5 rebranding.json . . . 69
A.5.1 Options that are handled differently . . . 69

B Appendix: Upgrading 7.5.* to 7.5.2 (Linux) **70**

1. Introduction

MedDream DICOM Viewer is a HTML based package for PACS server which is designed to aid professionals in every day's decision making process, connecting all the medical data into a unified and fast performing network. MedDream ensures a fast and reliable way to search, present and analyze the medical data (images and video files) on various devices: computers, smart phones, tablets and so forth.

MedDream covers: radiology, cardiology, oncology, gastroenterology and many other fields of medical application. It seamlessly integrates with various medical imaging devices, such as: ultrasound (US), magnetic resonance (MRI), positron emission tomography (PET), computed tomography (CT), endoscopy (ES), mammography (MG), digital radiography (DR), computed radiography (CR) ophthalmology, and so forth.

Core MedDream DICOM Viewer uses are:

- Replacement of hard copies, e.g. film archives, paper documents, etc.
- Remote access. MedDream provides a possibility to be mobile and work from any place in the world where the Internet is accessible. More than one person can access and view medical records at one time. Such functionality speeds up the collaboration among the professionals. So that a doctor in the hospital and a doctor that is in the different location may view the medical data and discuss about it simultaneously. The patient's medical history, various studies and images are found much faster comparing to the conventional paper-based methods.
- MedDream can be used as a standalone WEB Viewer or integrated into MedDream PACS, PacsOne PACS, dcm4chee Archive, Conquest, ClearCanvas, Orthanc PACS systems. Moreover, MedDream can be adapted to client's PACS system and easily integrated into RIS/HIS workflow.
- MedDream has multiple functions such as search of studies, viewing, analyzing, saving, exporting, forwarding images and videos, etc.

Note: In 7.5.2, some functionality of HIS integration is not ported to Java yet. Unfortunately the PHP backend, where this would still work, is not available any more.

---

2. Minimal server side requirements

2.1. Minimal hardware requirements

Parameter	Requirement
Processor	2.33GHz or higher x64-compatible (2 CPU cores)
Memory	8+ GB
Hard drive	30 GB (RAID 1, RAID 5, RAID 10)
Network Interface	100 Mbit/s

2.2. Minimal memory requirements

Minimal memory requirements for the best performance of the software:

- 8 GB of RAM if you plan to open more than 800 images (CT & MRI, PET-CT).
- 12 GB of RAM for more than 1500 images (multi-slice CT & PET-CT).
- 16 GB of RAM for more than 3000 images (cardiac or functional imaging).

2.3. Minimal software requirements

For security-related configuration, see security-summary.

Note: Starting from 7.5, MedDream is effectively Java-only and contains an embedded web server – **Apache is no more needed**.

2.3.1. Supported operating systems

MedDream supports the following operating systems:

- Windows Server 2008 (32 bit and 64 bit) and newer
- Windows 7 (32 and 64 bit) and newer
- Linux (64 bit)

Under 32-bit Linux there is no support for some types of images.

2.3.2. Supported PHP scripting interpreters

Since 7.5.1 MedDream does not need PHP by itself. However PacsOne does, and MedDream provides a file applet.php for integration into its web interface.

This module requires the cURL extension and is regularly tested under PHP 5.4 . . . 7.1 (newer ones might also work).

2.3.3. Supported Java runtime

Only Java 8 is supported and was tested.

- We recommend OpenJDK 8. The entire JDK is not needed, a JRE is sufficient. Builds for Windows are harder but still possible to obtain.
- Java 8 from Oracle is the traditional choice but less affordable as of now.

2.3.4. Decommissioning of MedDream

A particular version of MedDream is supported for a limited time (see the Info window or User Manual). When support is no longer provided, we strongly recommend to update to a supported version. Be aware that the customer is responsible for outcome (including, but not limited to, personal data protection) from operating a not supported version. Not supported software versions are used on customer's own risk.

3. Integration with PACS

Support for various PACSes is implemented in form of plugins. Their .jar files are expected in the same directory. They have no version numbers in the name in order to prevent hard-to-troubleshoot situations where multiple versions of the same plugin are present. Currently, if there is a need to know the version number of the plugin, then one can open the JAR file as a ZIP file and look for file META-INF/MANIFEST.MF; it will contain, among other things, the `Bundle-Version` property.

```
Bundle-Version: 2.0.10.SNAPSHOT
```

3.1. Integration modes

MedDream can access studies from PACS using:

- direct access to PACS database
- Web-based and other APIs
- DICOM 3.0 native interface (Query/Retrieve)
- direct file system access where a PACS is not necessary. (However for users' convenience some third-party user interface will still need to pass file/directory names to MedDream.)

Commercial integration solutions are also possible on request. In that case documentation with confidential information will be available as separate versions of this Manual.

For best performance, it is recommended to use direct access to PACS database whenever available.

3.1.1. Direct access mode on a different host

Warning: In the Direct access to PACS database mode, MedDream must either:

- be installed on the same host as PACS, or
- have access to studies' files, and likely path remapping needs to be configured.

Some databases contain paths to DICOM files stored on the PACS host (not on some common network storage, etc) and it is not always possible to create identical mount points or network drive letters on the MedDream host.

For this case most PACS plugins support a setting named *mappedStorageLocation* (or similarly) that specifies substrings in the path to be replaced with different substrings.

An example with PacsOne plugin: `...].mappedStorageLocation=/volume1/DICOMNL\=O:\|/volume2/DICOMNL\=P:\`. Here `/volume1/DICOMNL\` is replaced with `O:\` under which the network resource shared by a Linux system is mounted on a Windows system; in the same fashion `/volume2/DICOMNL\` is replaced with `P:\`. For best results, please take a look at Java logs for original paths.

Due to limitations of Windows, a service might fail to access a network drive mapped in an interactive session (even under the same user, even with the "Enable to interact with desktop" turned on). In this case, avoiding

a drive letter should give better results – just remap PACS-local directories to CIFS paths that would otherwise be attached to a drive letter (`...].mappedStorageLocation=/volume1/DICOMNL\=\server\share\DICOMNL/`).

3.1.2. Saving measurements and Key Objects

Since 7.1, to save Key Objects and Presentation States (annotations), the following parameters must be added to the configuration of every PACS integration plugin (except the “DICOM” plugin):

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].storeScpIp=127.0.0.1`

  IP of the Storage Server.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].storeScpPort=104`

  Storage Server Port.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].storeScpAet=PACSONE`

  Storage Server AE Title.

The local AE Title is configured by `com.softneta.meddream.dcmsnd.bind` as before.

3.1.3. Multiple PACSes in parallel

With most plugins multiple PACSes can be defined simultaneously, then used in the Search window to open studies (also called “Multi-PACS setup”).

This, however, is currently not supported in case of legacy HIS integration *without token validation*. Multiple instances of “QR” plugin (DICOM mode) are also not supported in some configurations due to a design bug.

Warning: Opening the same study from different PACSes *at the same time* is not supported. The frontend might behave erratically and you won’t be able to tell from which PACS a particular study is opened.

3.2. Direct access to PACS database mode

3.2.1. PacsOne

In case of PacsOne Server (http://pacsone.net/), MedDream implements direct database access mode.

A similar configuration called “Standard DB integration” uses the same plugin and a minimalistic database structure unrelated to the real PacsOne Server; the database is to be filled by custom solution implemented by the customer. Please contact info@softneta.com for example scripts etc.

3.2.1.1. PacsOne notes

applet.php for MedDream is not included in the installation archive any more. **To obtain it, please contact support@softneta.com.**

applet.php is regularly tested with PHP versions from 5.4 to 7.1, though newer ones might also work. It requires the php_curl PHP module.

Warning: PacsOne and mD Java Core process must run as the same user because newer versions of PacsOne create subdirectories with permissions too strict for different users. Please ensure that on Linux operating systems PacsOne and mD Core services use the same user or belong to the same group.

Warning: PacsOne must store the received DICOM files in the "DicomPart10" format. The other format, "Native", is not fully supported by MedDream (and you might have problems with other DICOM-related software). This setting is chosen during installation and later can be verified as follows:

- Windows: in the registry, REG_SZ `StorageFormat` under `HKEY_LOCAL_MACHINE\SOFTWARE\RainbowFish Software\PacsOne\${AeTitle}`;
- Linux: the setting `StorageFormat` in file(s) *.ini near PacsOne.exe.

PacsOne doesn't have an index on series.modality. If searches with a particular modality are unacceptably slow compared to the "All" choice, then you'll need to use the following MySQL command:

```
ALTER TABLE series ADD INDEX (modality);
```

Until 6.0, the HIS integration by Patient ID was using a certain kind of fuzzy matching that includes coerced values of Patient ID. For example, /?patient=12345 will also list studies with Patient ID "12345[some_original_value]". Since 6.0 this is turned off by default. If you still need the legacy behavior, then configure the "PacsOne" plugin with `strictSearchIsEnabled=false`.

Since MySQL 5.7, the ONLY_FULL_GROUP_BY mode is on by default. Consequently, MedDream's Search function uses a compatible and much slower query by default. If performance is paramount, we advise to disable ONLY_FULL_GROUP_BY and configure the "PacsOne" plugin with `useModalityAggregation=false`.

The new Reporting function is incompatible with PacsOne's "studynotes" and "attachment" tables. It is no more possible to use PacsOne for accessing reports created in MedDream, and vice versa.

3.2.1.2. MedDream configuration for PacsOne

1. Perform steps in the *Essential configuration and the first run* chapter.
2. If you intend to reuse PacsOne user management, then configure this kind of authentication in `application.properties` file:

```
spring.profiles.include=auth-pacsone
authorization.defaultLoginPermissions=SEARCH,PATIENT_HISTORY,UPLOAD_DICOM_
↪LIBRARY,3D_RENDERING
#authorization.users[0].userName=user1
#authorization.users[0].role=DOCUMENT_VIEW
authentication.database.databaseName=DB_NAME
authentication.database.jdbcUrl=jdbc:mysql://DB_HOST:DB_PORT/DB_NAME
```

   (for example, `jdbc:mysql://127.0.0.1:3306/${authentication.database.databaseName}` – we encourage placeholders to minimize human errors).

   The defaultLoginPermissions setting globally covers permissions not available in PacsOne user management. Also please remember that PacsOne's "Export" privilege is mapped to both

EXPORT_ISO and EXPORT_ARCH. The users[] setting can apply permissions to individual users.

Every user will automatically have either REPORT_VIEW or REPORT_UPLOAD permission, depending on the PacsOne "upload" privilege. It is not possible to have users without access to Reporting.

A different authentication method can still be configured according to *Built-in authentication and authorization*.

Note: By default, application.SAMPLE.properties enables auth-inmemory and this is done on another line. There is no error message if you assign to spring.profiles.include multiple times, and the outcome might be counter-intuitive. Make sure to comment the unneeded line out.

---

3. Edit `application.properties` file and update the options related to PacsOne plugin. The documentation below is for plugin version 2.0.11.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=PacsOne`

     Use this specific value of `type` when connecting to PacsOne.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=PacsOneRouter`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple PacsOne instances by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:mysql://127.0.0.1:3306/PACS_DB_NAME`

     JDBC connection string. Please note that Oracle is not supported.

     Note: SSL requirements can be disabled by adding the following arguments:

     - **verifyServerCertificate=false** Disables server certificate verification.
     - **useSSL=false** Disables SSL usage.
     - **requireSSL=false** Disables the SSL requirement.

     The plugin also supports PostgreSQL (`jdbc:postgresql://127.0.0.1:5432/DB_NAME`) for the "Standard DB integration" cases.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserName`

     Username for connecting to the database.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

     Password for connecting to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].mappedStorageLocation=D:/DICOM=Y:\\DICOM1|E:/DICOM=Y:\\DICOM2`

  (optional) Defines replacement paths for MedDream installed on a different host than the PACS and accessing images through mapped network drives.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].useModalityAggregation=true`

  (optional) Defines if modalities will be aggregated in database query results. Aggregation ensures that all modalities are visible in the Modality column but this could cause performance degradation. If set to false, then SQL mode must *not* contain 'ONLY FULL GROUP BY'. More information about SQL mode can be found at: https://dev.mysql.com/doc/refman/8.0/en/sql-mode.html.

  `false` is incompatible with PostgreSQL ("Standard DB integration") and will result in errors.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].strictSearchIsEnabled=`

  (optional) `true` forces exact matching by Patient ID and Accession Number everywhere, including the Search window. `false` forces wildcard matching everywhere, including HIS integration by Patient ID or Accession Number. `null` or unset is the default: exact matching for Patient History and HIS integration, wildcard matching for Search window.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].defaultStartDate=20181107`

  (optional) Default "from" date in *YYYYMMDD* format when empty in the user interface.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].defaultEndDate=20181108`

  (optional) Default "to" date in *YYYYMMDD* format when empty in the user interface.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].encoding=ISO-8859-1`

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].encodingTo=ISO-8859-8`

  (optional) Specify encoding of database strings when it's not UTF-8.

  - `encoding=ISO-8859-1`, `encodingTo=`:
    - search query is converted from UTF-8 to ISO-8859-1
    - search results are converted from ISO-8859-1 to UTF-8
  - `encoding=ISO-8859-1`, `encodingTo=ISO-8859-8`:
    - search results are converted from ISO-8859-1 to ISO-8859-8
    - search query is converted from ISO-8859-8 to ISO-8859-1
  - `encoding=`, `encodingTo=ISO-8859-8`:
    - search results are converted from UTF-8 to ISO-8859-8
    - search query is converted from ISO-8859-8 to UTF-8

  `encoding` and `encodingTo` are empty: UTF-8 is assumed everywhere (the outcome might still be influenced by JVM system encoding).

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method (PacsOne users in case of `spring.profiles.include=auth-pacsone`, etc).

3.2.1.3. Integration into PacsOne web interface

After you successfully log into MedDream and are able to access studies on PacsOne, the system can additionally be configured to open MedDream from PacsOne web interface.

Warning:

- All users of PacsOne will have the same set of MedDream permissions, configurable in `USER_PRIVILEGES` (applet.php) or by `authorization.defaultHisPermissions` (application.properties). If some users must not have access to, say, Forward or Reporting, then this restriction is necessary for everyone.

1. Extract the directory "token-service" from installation archive into a separate directory on the server (example result: /opt/token-service/token-service.xml). From now on it will be called installation directory of the Token Service.
2. Make a backup of PacsOne's php/applet.php.
3. Copy and Replace MedDream's applet.php over the PacsOne version of the file.
4. Verify whether `MEDDREAM_URL`, and especially `MEDDREAM_STORAGE_ID`, at the beginning are correct. `SSO_VALIDATOR_URL` will need correction, too, if the Token Service is configured with a different port.
   - If you added MedDream to a legacy PacsOne installation and Apache is proxying it as `/meddream`, then MEDDREAM_URL is good already.
   - If MedDream is running on a different port without proxying, then MEDDREAM_URL can include the full address, like `//localhost:8080/`.
   - If you are using MedDreamPACS, which offers MedDream at `/` and the PACS web interface at `/Pacs`, then just use `/` for MEDDREAM_URL.
5. Start the Token Service from a separate console window: change to its installation directory, then run `java -jar token-service.jar`. Wait until the message "Application ... started" appears.
6. In the PacsOne web interface, find some studies, tick at least one checkbox and click the "Show" button.

Note: Default values of `security.frameOptionsPolicy` and `security.contentSecurityPolicy` disallow embedding MedDream in an IFRAME, which is used by applet.php. The browser is refusing to display the embedded viewer and its console shows an error.

The quickest initial solution is to disable both in MedDream's application.properties:

```
security.frameOptionsPolicy=NONE
```

Afterwards you can experiment with `security.frameOptionsPolicy=ALLOW-FROM` and `security.frameOptionsWhitelist=<server IP address or hostname>`.

The Token Service has its own application.properties, not relevant to this issue.

Note: The "Show" button will appear on patient, study, series and image levels, however MedDream 7.5+ supports the study level only and PacsOne doesn't allow to hide the button on some levels.

For later convenience the token validator should run as a service. The installation directory already provides helper files token-service.debian, token-service.redhat, token-service.xml, token-service-jar-wrapper.*, token-service.NET?.exe, a separate version of application.properties. Please follow the *Running as a service* chapter and do not forget to use the corresponding file names.

Warning: It is recommended to remove or rename PacsOne's risdirect.php. Our applet.php is not designed for it, will not work in such a scenario and this combination might cause a slight security risk.

3.2.2. DCM4CHEE

For DCM4CHEE (https://www.dcm4che.org/) MedDream implements the direct database access mode.

3.2.2.1. DCM4CHEE notes

Warning: For DCM4CHEE 5, the 5.6 is our main testing platform. Some newer versions are tested occasionally. Support for 4.x was not ported from PHP due to very limited demand.

For DCM4CHEE 2, compatibility is regularly tested only with DCM4CHEE 2.4.17.

3.2.2.2. MedDream configuration for DCM4CHEE 5.x

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to DCM4CHEE 5.x plugin. The documentation below is for plugin version 2.1.4.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=Dcm4chee5`

     Use this specific value of `type` when connecting to DCM4CHEE 5.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=MainArchive`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple DCM4CHEE instances by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:mysql://127.0.0.1:3306/PACS_DB_NAME`

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:postgresql://127.0.0.1:5432/PACS_DB_NAME`

     JDBC connection string. Supported DBMSes are MySQL and PostgreSQL.

     In case of connection problems it might be necessary to add some more parameters to this string: `useJDBCCompliantTimezoneShift=true` for a time zone shift, `serverTimezone=UTC` for a specific timezone, `useSSL=false` to disable SSL, etc. (The first parameter-value pair is preceded by "?", the remaining pairs – by "&".) See https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-reference-configuration-properties.html for MySQL, or https://jdbc.postgresql.org/documentation/head/connect.html for PostgreSQL.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserName`

  Username used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

  Password used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].encoding=ISO-8859-1`

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].encodingTo=ISO-8859-8`

  (optional) Specify encoding of database strings when it's not UTF-8.

  - `encoding=ISO-8859-1`, `encodingTo=`:
    - search query is converted from UTF-8 to ISO-8859-1
    - search results are converted from ISO-8859-1 to UTF-8
  - `encoding=ISO-8859-1`, `encodingTo=ISO-8859-8`:
    - search results are converted from ISO-8859-1 to ISO-8859-8
    - search query is converted from ISO-8859-8 to ISO-8859-1
  - `encoding=`, `encodingTo=ISO-8859-8`:
    - search results are converted from UTF-8 to ISO-8859-8
    - search query is converted from ISO-8859-8 to UTF-8

  `encoding` and `encodingTo` are empty: UTF-8 is assumed everywhere (the outcome might still be influenced by JVM system encoding).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].aggregateDataByDatabase=true`

  (optional) Different approaches to collect distinct series-level values of *Modality* and *Source AET* for every study in the search results. `false` might improve performance in some cases (was not tested extensively).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].storage=fs1=c:\\mnt\\NAS1;fs2=c:\\mnt\\NAS2`

  Filesystem root directories, as configured in the LDAP tree via entries named "dcmStorageID". The plugin doesn't read this configuration from LDAP at the moment.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].pacsVersion=5.7`

  (optional) Two-component DCM4CHEE version number that defines what database fields are available. Namely, *Number of Frames* was added in 5.7 and this version is expected by default; if your PACS is older, you'll need to adjust this parameter to avoid SQL errors.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method.

3.2.2.3. MedDream configuration for DCM4CHEE 2.x

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. If you intend to (partially) reuse DCM4CHEE user management, then configure this kind of authentication in `application.properties` file:

```
spring.profiles.include=auth-dcm4chee
authentication.database.username=APP_USER
authentication.database.password=APP_PASS
authentication.database.jdbcUrl=jdbc:mysql://DB_HOST:DB_PORT/DB_NAME
authorization.defaultLoginPermissions=EXPORT_ARCH,EXPORT_ISO,FORWARD,VIEW_
↪PRIVATE,DOCUMENT_VIEW,3D_RENDERING
authorization.remapRoles.Doctor=SEARCH
authorization.remapRoles.WebAdmin=ADMIN
authorization.users[0].userName=user
authorization.users[0].role=UPLOAD_DICOM_LIBRARY,REPORT_UPLOAD
```

   `.jdbcUrl` supports not only MySQL but also PostgreSQL (`jdbc:postgresql://DB_HOST:DB_PORT/DB_NAME`) and MS SQL Server (`jdbc:sqlserver://DB_HOST:DB_PORT;database=DB_NAME`). `.username` and `.password` are credentials of an application user for reading the database tables; you can use the same credentials for the plugin below.

   `.defaultLoginPermissions` lists common permissions. In this example, the DCM4CHEE's predefined "Doctor" role additionally gets SEARCH, and the predefined user "user" gets UPLOAD_DICOM_LIBRARY and REPORT_UPLOAD. Because "user" belongs to "Doctor" by default, it now has UPLOAD_DICOM_LIBRARY, SEARCH and everything from defaultLoginPermissions. Similarly, the predefined user "admin" who has "WebAdmin" and "Doctor" roles in DCM4CHEE, now has ADMIN, SEARCH, etc (except UPLOAD_DICOM_LIBRARY and REPORT_UPLOAD).

   The full list of permissions can be found in *Built-in authentication and authorization*. That chapter also describes a few universal authentication methods.

   Note: By default, application.SAMPLE.properties enables auth-inmemory and this is done on another line. There is no error message if you assign to spring.profiles.include multiple times, and the outcome might be counter-intuitive. Make sure to comment the unneeded line out.

   ---

4. Edit `application.properties` file and update the options related to DCM4CHEE 2.x plugin. The documentation below is for plugin version 2.2.2.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=Dcm4chee2`

     Use this specific value of `type` when connecting to DCM4CHEE 2.x.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=RouterPacs`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple DCM4CHEE instances by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

  (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:mysql://127.0.0.1:3306/PACS_DB_NAME`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:postgresql://127.0.0.1:5432/PACS_DB_NAME`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:sqlserver://127.0.0.1:1433;database=PACS_DB_NAME`

  JDBC connection string. Supported DBMSes are MySQL, PostgreSQL and Microsoft SQL Server.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserName`

  Username used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

  Password used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].defaultStoragePath=C:/DCM4CHEE_INSTALL_DIR/server/default`

  Base directory for relative paths. By default all paths are relative to a certain location below the installation directory (see the example), unless one sets up an absolute filesystem prefix via addRWFileSystem() in JMX Console.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].charset=latin1`

  (optional) Defines the character set used in the database. If not empty, conversion from this character set to UTF-8 is attempted; otherwise the data is passed as is (default). `utf8` might work in case of double encoding.

  Note: The PACS GUI (/dcm4chee-web, /dcm4chee-web3, etc) is able to correctly display data even if the database server was configured incorrectly before sending the images and therefore parts of the database accessed by the plugin now contain garbled text or question marks (information is permanently lost). *DCM4CHEE has a certain workaround against this situation; there is no equivalent of it in the plugin.*

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].enableStatistics=false`

  (optional) `true` enables query statistics such as query time and so on. You might also want to set `logging.level.org.hibernate.stat=DEBUG` in application.properties to get more details.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].idleConnectionTestPeriod=900`

  (optional) Try lowering the value (in seconds) if the plugin doesn't reconnect automatically after the DBMS server is restarted (or the server closes the connection for being idle too long). A too low value will impact performance, though.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].mappedStorageLocations=D:/DICOM=Y:\\DICOM1|E:/DICOM=Y:\\DICOM2`

  (optional) Defines replacement paths for MedDream installed on a different host than the PACS and accessing images through mapped network drives or network paths.

We recommend to configure defaultStoragePath with a ready to use value, instead of using a PACS-local value and adding a separate replacement rule here. Of course the corresponding folder must be shared on the PACS, too.

- Example: `logging.level.com.mchange.v2.resourcepool.BasicResourcePool=WARN`

  Without this setting the plugin currently reports database connection problems with a too vague and misleading error message.

5. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method.

3.2.3. ClearCanvas

For ClearCanvas (https://clearcanvas.ca/) MedDream implements the direct database access mode.

3.2.3.1. ClearCanvas notes

Support was tested with ClearCanvas 2.0 and 13.2.

As IIS occupies the port 80, MedDream will need to run on a different port (the `server.port` setting). If MedDream URL looks "not nice" because of that, then you can use IIS as a reverse proxy; please contact support@softneta.com for instructions.

3.2.3.2. MedDream configuration for ClearCanvas

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to ClearCanvas plugin. The documentation below is for plugin version 1.2.4.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=ClearCanvas`

     Use this specific value of `type` when connecting to ClearCanvas.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=OldArchive`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple ClearCanvas instances by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:sqlserver://127.0.0.1:1433;database=ImageServer`

     URL should be specified in format: `jdbc:sqlserver://<HOST>\\<INSTANCE>:<PORT>;database=<DATABASE_NAME>`

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserName`

  Username used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

  Password used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].mappedStorageLocation=C:\\FS=Z:\\Studies`

  (optional) Defines pair of path strings that should be replaced in order to use mapped storages. Syntax for defining strings: “oldPathPart=newPathPart”.

  In case of multiple rules, separate them by pipe character and enter more specific ones first: `E:\\DICOM2=\\\\CLRCANVAS-SRV\\DICOM2\|E:\\DICOM=Z:\\DICOM`.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].strictSearchIsEnabled=`

  (optional) `true` forces exact matching by Patient ID and Accession Number everywhere, including the Search window. `false` forces wildcard matching everywhere, including HIS integration by Patient ID or Accession Number. `null` or unset is the default: exact matching for Patient History and HIS integration, wildcard matching for Search window.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1:MEDDREAM_PORT/`. Use login credentials that are valid for chosen method.

3.2.4. Conquest

For Conquest (https://ingenium.home.xs4all.nl/dicom.html) MedDream implements the direct database access mode.

3.2.4.1. Conquest notes

MedDream does not support the proprietary “V2 (allows NKI compression)” image format. All images that were received by Conquest with this setting on, including those of the example patient “HEAD EXP2”, will be unusable.

3.2.4.2. MedDream configuration for Conquest

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to Conquest plugin. The documentation below is for plugin version 1.1.1.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=Conquest`

     Use this specific value of `type` when connecting to Conquest.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=ResearchPacs`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple Conquest instances by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

(optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

  (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:mysql://127.0.0.1:3306/PACS_DB_NAME`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:postgresql://127.0.0.1:5432/PACS_DB_NAME`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].url=jdbc:sqlserver://127.0.0.1:1433;database=PACS_DB_NAME`

  JDBC connection string. Supported DBMSes are MySQL, PostgreSQL, Microsoft SQL Server, dBASE 3.

  MS SQL server isn't directly supported by Conquest. This is a special case when Conquest uses an ODBC source backed by the SQL Server, and MedDream connects to the server directly.

  Please contact support@softneta.com if you would like to use dBASE 3 integration.

  In case of connection problems it might be necessary to add some more parameters to this string, like `useSSL=false` to disable SSL. (The first parameter-value pair is preceded by "?", the remaining pairs – by "&".) See https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-reference-configuration-properties.html for MySQL, or https://jdbc.postgresql.org/documentation/head/connect.html for PostgreSQL, or https://docs.microsoft.com/en-us/sql/connect/jdbc/setting-the-connection-properties?viewFallbackFrom=sql-server-previousversions for Microsoft SQL Server.

  The driver parameters `useJDBCCompliantTimezoneShift=true`, `serverTimezone=UTC` and `autoReconnect=true` are added automatically.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserName`

  Username used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

  Password used to connect to the database.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].pathToDicomIni=E:\\Conquest\\dicom.ini`

  Full path to the Conquest configuration file dicom.ini. MedDream will read `MAGDeviceXXX=...` lines from it.

  There is no "mappedStorageLocation" configuration option like in other plugins. In case when MedDream is on a different host, one can copy dicom.ini to a file on MedDream host, leave only MAGDeviceX lines and update them with shared paths that are accessible on MedDream host. For example, `MAGDevice1=E:\DICOM` might change to `MAGDevice1=\\conquest-srv\dicomstorage1`.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method.

3.3. Integration over Web or other APIs

3.3.1. Orthanc

For Orthanc (https://www.orthanc-server.com/) MedDream uses its custom Web API.

Direct database access wasn't implemented as it would likely offer little performance gain due to database schema complexity. You might try the DICOM mode as an alternative.

3.3.1.1. Orthanc notes

The API is not used for authentication because tested versions of Orthanc offer only global credentials (for a single user). These credentials must still be entered in plugin configuration, though.

3.3.1.2. MedDream configuration for Orthanc

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to Orthanc plugin. The documentation below is for plugin version 2.0.8.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=Orthanc`

     Use this specific value of `type` when connecting to Orthanc.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=Telemedicine`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple Orthanc instances by using different values of `id` and remaining options.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].baseUrl=http://127.0.0.1:8042`

     Base URL that points to the root of Orthanc Web API. (Endpoints `/tools/find`, `/instances` etc will be appended automatically.)
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserLogin`

     Username for Basic Authentication.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

     Password for Basic Authentication.
   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].dicomCacheDirectory=${com.softneta.meddream.tempDir}/dcm/Telemedicine`

(optional) A directory for caching images. Without this setting a file is downloaded anew every time and it's not possible to examine the contents for troubleshooting.

This plugin supports URIs (starting with `file://`). Making them relative to tempDir or another setting is encouraged to minimize human errors.

Multiple instances of this plugin should have different directories in order to not mix up potentially different versions of the file with the same Study/Series/SOP Instance UIDs.

As of MedDream 7.5.1+, this directory is not cleaned automatically and you need to configure a separate entry under `com.softneta.tempCleaner.tempItems[]` (see *Temporary files cleaner*).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].strictSearchIsEnabled=true`

  (optional) `true` forces exact matching by Patient ID and Accession Number everywhere, including the Search window. `false` forces wildcard matching everywhere, including HIS integration by Patient ID or Accession Number. `null` or unset is the default: exact matching for Patient History and HIS integration, wildcard matching for Search window.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method.

3.3.2. DICOMweb

Universal HTTP-based communication over RESTful protocols like QIDO-RS and WADO-RS, and the legacy WADO-URI protocol.

3.3.2.1. DICOMweb notes

WADO-RS does not provide means to request a particular set of attributes. Study metadata from a server inclined to return all attributes can be enormous. Some non-standard parameter, like "dataset=meddream", could be implemented in the server to support optimal and reduced attribute sets at the same time. Or, the server could be configured for a reduced set if MedDream is the sole client.

The DICOM Standard does not offer an attribute for the date when the study was received by the PACS. Similarly there is some confusion regarding which attribute should indicate Application Entity Title of the sender. As a result, typical PACSes do not return this data and MedDream search columns "Received On" and "Source AE" remain empty. If you'll find out what tags a particular PACS uses, then they can be configured via plugin options `studyReceivedDateTag` and `sourceAeTitleTag`.

The plugin supports the following authentication choices:

- no authentication,
- HTTP Basic Authentication,
- a custom HTTP POST to a configurable address that accepts HTML FORM variables "login" and "password", and returns HTTP 200/202 with a configurable cookie. This cookie is included in all subsequent requests and automatically refreshed when needed.

When configuring a custom query in URLs, do not attempt to override some parameters added by the plugin dynamically; as the plugin does not check for duplicates, the server will likely accept the second occurrence (the duplicate from the plugin) instead.

3.3.2.2. MedDream configuration for DICOMweb

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure a universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to DICOMweb plugin. The documentation below is for plugin version 3.0.0; examples are from integrations with DCM4CHEE 5.12, Orthanc 1.5.8 and PacsOne 6.6.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=DICOMweb`

     Use this specific value of `type` when connecting via DICOMweb.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=MainArchiveDW`

     Identifies the plugin/configuration pair in the search window drop-down menu. You can connect to multiple PACSes by using different values of `id` and remaining options.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

     When targeting an older PacsOne installation that supports WADO-URI only, you can use `false` here and choose a different plugin (`PacsOne` or even `QR`) with same `id` and imageApiEnabled=false as the counterpart.

     Speaking of any PACS, this plugin's Image API can be used as an alternative if sharing the DICOM files over network is somehow not possible and slower operation is not an issue.

     Note: An equivalent of the legacy `$pacs = 'WADO'` integration can be constructed from the QR plugin (Search API) and this plugin (Image API).

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].qidoRsUrl=http://127.0.0.1:8080/dcm4chee-arc/aets/DCM4CHEE/rs?fuzzymatching=true`

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].qidoRsUrl=http://127.0.0.1:8042/dicom-web`

     Base URL of QIDO-RS API. Can include a query for additional or non-standard parameters. (The endpoint `/studies` and relevant dynamic parameters will be appended automatically.)

     In this example, DCM4CHEE 5 requires the "fuzzymatching" parameter or else searching by Patient Name is not possible at all.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoRsUrl=http://127.0.0.1:8080/dcm4chee-arc/aets/DCM4CHEE/rs`

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoRsUrl=http://127.0.0.1:8042/dicom-web`

     Base URL of WADO-RS API. Can include a query for additional or non-standard parameters. (The endpoint `/studies` will be appended automatically.)

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoUriUrl=http://127.0.0.1:8080/dcm4chee-arc/aets/DCM4CHEE/wado`

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoUriUrl=http://127.0.0.1:8042/wado`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoUriUrl=http://127.0.0.1/wado.php`
- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].wadoUriUrl=http://127.0.0.1/wado.php?username=UserLogin&password=UserPassword`

  Base URL of WADO-URI API. Can include a query for additional or non-standard parameters. (The relevant dynamic parameters will be appended automatically.)

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].username=UserLogin`

  (optional) Username for Basic Authentication or custom form authentication.

  Not needed for default configuration of DCM4CHEE 5. Definitely needed for Orthanc with AuthenticationEnabled=true. Needed for PacsOne by default unless credentials are included in the URL (less secure).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].password=UserPassword`

  (optional) Password for Basic Authentication or custom form authentication.

  Not needed for default configuration of DCM4CHEE 5. Definitely needed for Orthanc with AuthenticationEnabled=true. Needed for PacsOne by default unless credentials are included in the URL (less secure).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].loginUrl=http://server/path`

  (optional) URL for a custom form authentication. If empty, then Basic Authentication or no authentication (depending on whether both username and password are configured) is used instead.

  Obviously not applicable to DCM4CHEE, Orthanc or PacsOne.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].loginCookie=MySessionCookie`

  (optional) Cookie name for a custom form authentication. Default value is “suid”.

  Obviously not applicable to DCM4CHEE, Orthanc or PacsOne.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].sourceAeTitleTag=19088743`

  (optional) Tag number, likely PACS-specific, to use for Source AE Title in search results. Default value is zero, which means disabled.

  Must be in decimal notation, e. g., (0123,4567) => 0x01234567 => 19088743.

  To our knowledge neither DCM4CHEE or Orthanc support such an attribute.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].studyReceivedDateTag=19106219`

  (optional) Tag number, likely PACS-specific, to use for Study Received Date in search results. Default value is zero, which means disabled.

  Must be in decimal notation, e. g., (0123,4567) => 0x01234567 => 19088743.

  To our knowledge neither DCM4CHEE or Orthanc support such an attribute.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].multipleModalitySupport=true`

  (optional) `false` will yield an error message when the user selects multiple modalities. Use this if the PACS does not support multiple modalities and search results confuse the users.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].dicomCacheDirectory=${com.softneta.meddream.tempDir}/dcm/MainArchiveDW`

  (optional) A directory for caching images. Without this setting a file is downloaded anew every time and it's not possible to examine the contents for troubleshooting.

  This plugin supports URIs (starting with `file://`). Making them relative to tempDir or another setting is encouraged to minimize human errors.

  Multiple instances of this plugin should have different directories in order to not mix up potentially different versions of the file with the same Study/Series/SOP Instance UIDs.

  As of MedDream 7.5.1+, this directory is not cleaned automatically and you need to configure a separate entry under `com.softneta.tempCleaner.tempItems[]` (see *Temporary files cleaner*).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].strictSearchIsEnabled=false`

  (optional) `true` forces exact matching by Patient ID and Accession Number everywhere, including the Search window. `false` forces wildcard matching everywhere, including HIS integration by Patient ID or Accession Number. `null` or unset is the default: exact matching for Patient History and HIS integration, wildcard matching for Search window.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for chosen method.

3.4. DICOM mode

MedDream is able to communicate over DICOM 3.0 native interface with any PACS that supports Query/Retrieve.

3.4.1. DICOM mode notes

The Query/Retrieve SCU identifies itself as "MEDDREAM" by default. (Do not forget to configure the PACS to accept connections from this AET.) This can be changed by the `localAET` setting of the plugin. Binding to a particular port is usually not required, and the current Java implementation doesn't support that.

The plugin also contains its own Store SCU that is used to send annotations and Key Object Selection documents back to the PACS. (With other plugins, a common DICOM SCU in Java Core is used, see *Saving measurements and Key Objects*.) As a result,

- its Calling AE Title is also configured by the `localAET` plugin setting, instead of com.softneta.meddream.dcmsnd.bind;
- the plugin-level settings `storeScpIp`, `storeScpPort` and `storeScpAet` (for the common DICOM SCU) are not needed – the plugin uses the same `remoteHost`, `remotePort` and `remoteAET` as for Query/Retrieve.

The local Store SCP is implemented not in the plugin but in Java Core. It **must** bind to a port (we recommend 11116) and listens on all available addresses. The mandatory settings are:

- `com.softneta.dicomStoreService.localAETitle`,
- `com.softneta.dicomStoreService.port`,
- `com.softneta.dicomStoreService.acceptAETitles` – known Calling AE Titles,
- `com.softneta.dicomStoreService.saveURIDirectory` – base directory for received files.

3.4.2. MedDream configuration for DICOM mode

1. Perform steps in the *Essential configuration and the first run* chapter.
2. Choose and configure some universal authentication method according to *Built-in authentication and authorization*.
3. Edit `application.properties` file and update the options related to Query/Retrieve plugin. The documentation below is for plugin version 2.3.4.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=QR`

     Use this specific value of `type` when connecting to a PACS over the DICOM 3.0 protocol.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=AnonStudies`

     Identifies the plugin/configuration pair in the search window drop-down menu. Multiple instances of this plugin, however, are not recommended at the moment.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].searchApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair returns a study list for the Search window and provides study metadata. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].imageApiEnabled=true`

     (optional) Specifies if this plugin/configuration pair retrieves image data. Default is `true`.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].remoteAET=PACSONE`

     The AE Title of the remote machine.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].remoteHost=127.0.0.1`

     IP address or hostname of the remote machine.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].remotePort=104`

     Port number of the remote machine.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].localAET=MEDDREAM`

     The AE Title of the DICOM C-FIND/C-MOVE/C-STORE clients implemented in this plugin. Also used as the C-MOVE destination.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].cacheDir=${com.softneta.dicomStoreService.saveURIDirectory}/PACSONE/`

     A directory with images that are received by *local C-STORE SCP* in background. *The trailing slash/backslash is mandatory.*

     The last subdirectory must be equal to `remoteAET`, as the local SCP adds one automatically to the configured base directory.

     This plugin supports URIs (starting with `file://`). Making them relative to receiver's saveURIDirectory is encouraged to minimize human errors.

   - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].timeout=90000`

     (optional) DICOM file receive timeout in milliseconds (starting from the C-MOVE request).

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].retryDelay=1000`

  (optional) DICOM file existence check period in milliseconds.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].strictSearchIsEnabled=true`

  (optional) `true` forces exact matching by Patient ID and Accession Number everywhere, including the Search window. `false` forces wildcard matching everywhere, including HIS integration by Patient ID or Accession Number. `null` or unset is the default: exact matching for Patient History and HIS integration, wildcard matching for Search window.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].multipleModalitySupport=true`

  (optional) `false` will yield an error message when the user selects multiple modalities. Use this if the PACS does not support multiple modalities and search results confuse the users.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].defaultStartDate=20181107`

  (optional) Default "from" date in *YYYYMMDD* format when empty in the user interface. This is for some PACSes that require the Study Date search key in every C-FIND request.

- Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].defaultEndDate=20181108`

  (optional) Default "to" date in *YYYYMMDD* format when empty in the user interface. This is for some PACSes that require the Study Date search key in every C-FIND request.

4. Restart Core for changes to take effect, and navigate to `http://127.0.0.1/`. Use login credentials that are valid for configured method.

3.5. File system access mode

This MedDream mode allows to view single DICOM files, or directories containing files (of a single study or multiple studies). In fact, no PACS is necessary.

However there is no search functionality. **A HIS (or equivalent application) must track studies/images together with corresponding paths, and offer hyperlinks to MedDream.**

3.5.1. File system mode notes

MedDream is opened with a relative path to a single file, or to a directory. Absolute paths are not allowed due to security concerns.

If a directory is specified, then DICOM files are collected directly below it by default. Number of scanned subdirectories can be increased by the `maxDepth` plugin option.

Warning: As of 7.5.2, there is only a limited support for copies of the same DICOM file (same Study/Series/SOP Instance UIDs) in different directories. For example, the frontend will behave erratically when such studies are opened at the same time. The Reporting function is not recommended at all: it will treat such studies as different ones, their reports might become mixed up after restarting the backend and reopening the studies again.

This plugin is the only one that supports a custom HIS integration parameter "file". Due to architectural shortcuts, it also supports the ubiquitous parameter "study". But, the latter is not intended for HIS integration; if used this way, it will work only for a short time (until a particular in-memory cache expires) after successfully opening studies via "file".

Do not set `searchApiEnabled` or `imageApiEnabled` in plugin configuration to `false`, as this plugin can't be combined with others, even with other plugin instances of the same type.

The login form is not needed to view images and can therefore remain disabled together with the Search window. However both features are needed to access the Settings dialog. For that one must log in interactively as a user with the "ADMIN" privilege.

3.5.2. MedDream configuration for File system mode

1. **Perform steps in the *Essential configuration and the first run* chapter.**
    1. Configure the HIS integration in `application.properties` file:

```
spring.profiles.include=auth-his
authentication.his.validHisParams=file
authentication.his.maxObjects=10
authorization.defaultHisPermissions=EXPORT_ISO,EXPORT_ARCH,FORWARD,
↪UPLOAD_DICOM_LIBRARY,DOCUMENT_VIEW
```

    The "maxObjects" setting allows to limit the number of studies that is allowed to open at once. If the directory specified in /?file=... contains more, the operation will fail. This prevents situations where too many objects would render the frontend unusable.

2. **Plan how MedDream settings will be changed:**
    - **(Option 1)** Choose and additionally configure some universal authentication method according to *Built-in authentication and authorization*, then add both "ADMIN" and "SEARCH" privileges to users. These users will be able to log in, ignore zero results in the Search window, then go to the Settings window. There is little sense in having users without the "ADMIN" privilege, due to absent studies in the Search window.
    - **(Option 2)** Add the "ADMIN" privilege to `authorization.defaultHisPermissions`. Remember that all users will then be able to change MedDream settings and might thus interfere with each other's work. As a short-time measure, however, this option might still be more attractive than the first one.
3. Edit `application.properties` file and update the options related to the plugin "FileSystem". The documentation below is for plugin version 3.0.0.
    - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].type=FileSystem`

      Always set this to `FileSystem`.
    - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].id=File-System-Plugin`

      Any alphanumeric string that identifies this plugin instance.
    - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].rootDirectory=C:\\dicoms\\files`

      Full path to the base directory below which the DICOM files can be found.
    - Example: `com.softneta.meddream.pacs.configurations[PLUGIN_INDEX_NUMBER].maxDepth=1`

      The number of subdirectories traversed when collecting files. Minimum allowed value and default value is 0.
4. Test the functionality by browsing to `http://127.0.0.1/?file=`*`PATH_TO_YOUR_FILE`*

    For example, the directory `C:\PACS\dcm4chee\server\default\archive` contains DCM4CHEE 2.x archive tree, with deeper levels named like `year\month\day\hour\...`. This directory is specified by `rootDirectory`. Then *`PATH_TO_YOUR_FILE`* could be `2012/2/4/0/43D7AA94/2569DF62/9242C40A`.

4. Image Access from hospital information system (HIS)

In MedDream 7.5.2, the legacy “unsafe” integration (an object identifier in the URL is sufficient to display it) supports a few filters.

- `http://localhost/?study=1.2.392.200036.9107.500.110113,1.2.840.114350.2.171.2.798268.2.567817820.1`

  Opens the specified study (or studies) at once. Multiple values can be separated with commas.

- `http://localhost/?accnum=0000000005`

  Opens **all studies** resolved by this single Accession Number **at once in the Viewer**. No multiple values.

- `http://localhost/?patient=TT054270`

  Opens **all studies** of this single patient **at once in the Viewer**. No multiple values.

  There is a risk that the number of studies is too large – for example, “0” is often assigned during anonymization and would therefore match a lot of studies. Such an operation will fail due to exceeded `authentication.his.maxObjects`.

  Currently there is no support for the legacy integration by Patient ID that used to result in a filtered Search window.

- `http://localhost/?patient=5702887&accnum=0000000005`

  Opens all studies resolved by this filter at once. No multiple values.

- `http://localhost/?file=DIR-WITH-MULTIPLE-STUDIES`
- `http://localhost/?file=STUDY-DIR`
- `http://localhost/?file=STUDY-DIR/102.dcm,295.dcm`

  Opens multiple studies, a single study or single image(s). Multiple values can be separated by commas; however MedDream will treat them as separate studies and won’t join images, even from adjacent files, into the same study.

  A single series is obviously possible if files are grouped into series subdirectories.

These parameters, in turn, are supported by almost all PACS plugins. Rare exceptions are listed under a particular PACS integration in *Integration with PACS* chapter.

Additional limitations:

- only HTTP GET method;
- only a single filter – for example, you can separately configure “study” or “patient”, but not “study” and “patient” at the same time for using either one or another, etc;
- multiple PACS plugins are not allowed (particularly due to unsupported parameter “storage” in the URL).

To enable this integration mode, add `auth-his` to the `spring.profiles.include` parameter, specify a single URL parameter (or their combination) in `authentication.his.valid-his-params`, maximum number of resulting studies in `authentication.his.maxObjects` and list permissions in `authentication.his.defaultHisPermissions`. For example,

```
spring.profiles.include=auth-pacsone,auth-his
authentication.his.valid-his-params=study
authentication.his.maxObjects=2
#authentication.his.useSameSession=true
authorization.defaultHisPermissions=EXPORT_ARCH,DOCUMENT_VIEW
```

```
spring.profiles.include=auth-pacsone,auth-his
authentication.his.valid-his-params[0]=patient,accnum
authentication.his.maxObjects=2
#authentication.his.useSameSession=true
authorization.defaultHisPermissions=EXPORT_ARCH,DOCUMENT_VIEW
```

(Formally the first example needs `...valid-his-params[0]=study`, too. Luckily "[0]" is optional in this situation.)

The "useSameSession" parameter allows to have multiple viewer windows during "unsafe" HIS integration. By default, a newly opened viewer window also replaces the session, therefore a window from the earlier session may get "Access denied" errors.

However it is recommended to **use the new token-based integration mode** instead:

- a token validator service (its reference implementation is bundled with MedDream for integration into PacsOne web interface) accepts object identifiers and returns a generated token. Multiple object identifiers and storage names are supported;
- MedDream HIS integration links include only a single parameter called "token";
- when MedDream loads, it obtains object identifiers (study, accession number, ...) from the token service.

For more information, including specification of the token service, see a separate document "MedDream DICOM Viewer integration MANUAL" (contact info@softneta.com if needed).

5. Deployment

This chapter is dedicated to PACS-independent part of the deployment. Normally you should start reading at support for a specific PACS under *Integration with PACS*, as those parts refer to this chapter anyway.

In case of Linux, this chapter assumes that the user has root permissions and appropriate part of MedDream installation archive was extracted into /opt/meddream (the file /opt/meddream/MedDream-*.jar exists).

5.1. Configuration

5.1.1. Essential configuration and the first run

The main configuration file is `application.properties` in the directory of MedDream-*.jar. After any changes it is required to restart the Java application.

Note: During an upgrade it's recommended to use application.SAMPLE.properties (shipped with MedDream) as a template and update it based on contents of your old configuration file. This will prevent misunderstandings like parameters renamed or removed between versions.

---

1. Revise the following most important configuration parameters:
    1. Application TCP/IP port

        ```
        server.port = APPLICATION_PORT
        ```

        Allows to choose a different port if the default one is occupied. Example: `server.port=80`.
    2. Under Linux, FFmpeg installation is required to create video thumbnails and convert some less-supported formats.

        ```
        com.softneta.thumbnails.ffMpegExecutable = PATH_TO_FFMPEG
        ```

The default value is a simple name of the executable, `ffmpeg`. In most cases it is enough under Linux.

It might be enough under Windows, too, if you have a recent FFmpeg installation and the directory is included in PATH environment variable. Otherwise update this parameter with a full path to the bundled `${com.softneta.meddream.configRoot}/sys/ffmpeg/ffmpeg.exe`; you might need to use a literal path without placeholders.

3. Remaining values

   During the first start these can usually remain unchanged. See *Reference for remaining configuration parameters*.

2. Add native libraries bundled with MedDream for a more comprehensive support of compressed DICOM transfer syntaxes:
   - open libs.zip and choose the folder suitable for your operating system. For example, `win-i686`;
   - extract file(s) from that folder to a new folder of your choice. For example, `C:\MedDreamPACS-Premium\services\lib`;
   - from now on you'll need to specify the path to this folder in the command that starts Java: `-Djava.library.path=PATH_TO_FOLDER`.
3. Start the Java application manually with a console command "java -jar", for example, `java -jar MedDream-7.5.2.jar`.

   After adding the native libraries mentioned above, the command changes accordingly. Furthermore, a certain problem with Reporting function requires the -DANTLR_USE_DIRECT_CLASS_LOADING option.

   A full example: `java -DANTLR_USE_DIRECT_CLASS_LOADING=true -Djava.library.path=C:\MedDreamPACS-Premium\services\lib -jar MedDream-7.5.2.jar`.

At this point it should just start up without errors, and the message "Application ... started" will eventually appear.

The login form at http://localhost:APPLICATION_PORT/ is disabled by default (results in a "403 Forbidden" message) unless *Built-in authentication and authorization* is configured. On the other hand, application.SAMPLE.properties includes "in-memory" by default, with a predefined password. As the optimal method might be PACS-dependent, you can configure it a bit later, together with the plugin for a particular PACS.

5.1.1.1. Additional steps for Linux

1. FFmpeg is required in order to display thumbnails of MPEG2 DICOM files and non-BD-compatible MPEG4 videos (Transfer Syntax UID 1.2.840.10008.1.2.4.102). If your distribution does not provide ffmpeg, avconv package may be used. In that case a symlink to avconv binary is required:

   ```
   ln -s /usr/bin/avconv /usr/bin/ffmpeg
   ```

2. To display non-BD-compatible MPEG4 videos (Transfer Syntax UID 1.2.840.10008.1.2.4.102), latest version of FFmpeg might be required.
3. If Java application will run under a non-root user, then adjust file system permissions:

   ```
   chown LIMITED_USER:LIMITED_GROUP -R /opt/meddream
   find /opt/meddream -type d -exec chmod 775 {} \;
   find /opt/meddream -type f -exec chmod 664 {} \;
   ```

   Running Java application as root is OK for initial tests, however in production environment it should run under some limited user instead, ideally the one who can access all DICOM files in read only mode. A compromise would be to run the JVM under the same user as the PACS process.

This step makes sure that a process running under a limited user can access its files, especially the ones created earlier under name of root. In the simplest case you can examine the log file after starting the process under a different user, as even the log file will likely be non-writeable.

5.1.2. System.json

The file system.json (sys/settings/system.json by default) is automatically created if missing when the Java application starts. Several key features require this file to be updated; the application must be restarted afterwards.

An example of the entire file is shown below. When updating, take care to preserve the syntax (quotes, commas, brackets, etc); a text editor with JSON syntax checking is recommended.

```
{
  "disableMultiFrameVideoAutoLoad": false,
  "threeDimensionServiceIntegration": {
    "port": "8089",
    "protocol": "https:",
    "serverIP": "127.0.0.1",
    "pathPrefix": "D:\\DICOM",
    "pathPrefixReplaceTo": "Z:\\DICOM"
  },
  "features": {
    "patientHistory": true,
    "keyObjectAndPresentationStateQuickSave": true,
    "keyObjects": true,
    "presentationState": true,
    "searchSettings": true,
    "viewerSettings": true,
    "reportSettings": true,
    "hangingProtocolsSettings": true,
    "export": true,
    "archive": true,
    "search": true,
    "reports": true
  },
  "blackListedSopClasses": [
    "1.2.840.10008.5.1.4.1.1.66",
    "1.2.840.10008.5.1.4.1.1.4.2"
  ],
  "forwardPacs" : [
    {
      "storeScpIp": "127.0.0.1",
      "storeScpPort": 104,
      "storeScpAet": "PACS",
      "description": "An example destination for Forward"
    }
  ],
  "languages" : ["lt","en","ru"],
  "dicomLibraryConfiguration": {
      "dicomLibrarySender": "someone@mail.com",
      "dicomLibrarySubject": "My special study"
  },
  "asynchronousStudyLoading": false,
  "formatIntegrationLinkInViewer": false
}
```

- disableMultiFrameVideoAutoLoad: false

  (reserved) Disables multiframe and video autoload. Used for automated testing purposes, do not change this setting.

- patientHistory: true

The viewer window will load patient history.

- keyObjects: true

  The viewer window will load key objects.

- keyObjectAndPresentationStateQuickSave: true

  The viewer window will display the Key Object And Presentation State Quick Save button.

- presentationState: true

  The viewer window will load presentation state objects.

- searchSettings: true

  Enable search settings page in the settings window.

- viewerSettings: true

  Enable viewer settings page in the settings window.

- reportSettings: true

  Enable report settings page in the settings window.

- hangingProtocolsSettings: true

  Enable hanging protocol settings page in the settings window.

- export: true

  Globally enable the Export/ISO (Burn) function. If disabled, it won't be available even to users who have the EXPORT privilege.

- archive: true

  Globally enable the Export/Archive (Save Active . . .) function. If disabled, it won't be available even to users who have the ARCHIVE privilege.

- search: true

  Globally enable or disable the Search window. If disabled, the window will be replaced with an error message even for users who have the SEARCH privilege.

  The window is enabled by default to make the installation simpler. However currently it implements no per-user filters and allows access to every patient on the PACS. Do not forget to disable it if not needed.

  For example, if MedDream will open only via HIS integration, `true` is still recommended as an *initial* choice, as interactive login usually provides a faster way to verify whether images can be displayed. In this case do not forget to switch to `false` after testing.

- reports: true

  Globally disable or enable the Reporting window. If disabled, it won't be available even to users who have the REPORT_VIEW or REPORT_UPLOAD privilege.

- blackListedSopClasses: ["1.2.840.10008.5.1.4.1.1.66"]

  Thumbnails of images with these SOP Classes will not be shown in the viewer.

  This list is appended to a hardcoded list of non-image objects from DICOM Standard. (By the way, the above . . . 1.1.66 SOP Class is already there.) Main purpose of this setting is to hide any Private SOP Classes not known in advance.

- storeScpIp: "127.0.0.1"

  Destination IP address for the Forward function.

- storeScpPort: 4242

  Destination TCP port for the Forward function.

- storeScpAet: “PACS”

  Destination AE Title for the Forward function.

- description: “An example destination for Forward”

  Human-readable description of a destination device for the Forward function.

- languages: [“lt”,”en”,”ru”]

  List of localization languages that will be offered (in this order) in MedDream user interface. The first entry becomes the default on user’s computer when MedDream is opened there for the first time.

- dicomLibrarySender: “someone@mail.com”

  The Share function requires a non-empty value. (If empty, 7.5+ displays the same “license does not allow updates” message even if the license is OK.)

- asynchronousStudyLoading: false

  (experimental) When a single plugin of type QueryRetrieve is used, you can change this to `true` and enable a faster mode based on websockets.

- formatIntegrationLinkInViewer: false

  (experimental) After the viewer opens, adjust the URL so that it becomes a HIS integration link suitable for copying. Only the “study” parameter and a single PACS plugin are supported.

5.1.3. Reference for remaining configuration parameters

The parameters listed below can be left alone in most situations. The rarest of them aren’t included in application.SAMPLE.properties for the sake of its simplicity.

1. MedDream root directory

   ```
   com.softneta.meddream.configRoot = PATH_TO_MEDDREAM_DIRECTORY
   ```

   All other directories are based on this value by default, therefore a correct specification means less parameters are needed in application.properties.

   There is no default value and the current directory is substituted automatically. The substitution is based on the “user.dir” Java property, which in turn must be initialized by the JVM and not via the -D option.

   If the outcome is still incorrect, you can try to specify configRoot explicitly. It is expected in the URI format; the “file://” prefix is mandatory and relative paths are not supported.

   Examples:

   ```
   com.softneta.meddream.configRoot = file:///opt/meddream
   com.softneta.meddream.configRoot = file:///C:/MedDreamPACS/
   MedDream
   com.softneta.meddream.configRoot = file:///C:/PROGRA~1/PacsOne/
   php/meddream
   ```

2. Path to MedDream temporary files

   ```
   com.softneta.meddream.tempDir = PATH_TO_MEDDREAM_TEMP_FILES
   ```

   Allows to specify the path to a location where MedDream temporary files are stored; the default “temp” below configRoot might not always be suitable.

   Examples:

`com.softneta.meddream.configRoot = ./temp` (non-URI format might not work in all cases)

`com.softneta.meddream.configRoot = file:///mnt/mdtmp`

`com.softneta.meddream.configRoot = file:///E:/mdtmp`

Note: Since 7.0 some cached files might need paths longer than 260 characters which are not allowed under Windows. The only solution is to have com.softneta.meddream.tempDir as short as possible.

---

3. Session duration in seconds

```
server.servlet.session.timeout = 60
```

Some PACS plugins are slower and might take several minutes to load even the structure of a large study (afterwards error icons are displayed instead of thumbnails). In these situations the timeout should be increased.

Please remember that due to larger timeouts, client connections will persist longer in case of inactivity or incorrectly closed MedDream window (using the “X” icon on the corner instead of “Logoff”). With a license that limits the number of concurrent connections, this might exhaust the free connections more often.

4. Session cookie name

```
server.servlet.session.cookie.name = MEDDREAMSESSID
```

Value in the example is a hidden default and won’t likely need correction.

For cases where the current value is used by HIS or different software on the same domain; in these situations MedDream unpredictably loses its session.

5. IFRAME embedding permissions

This is the default and allows embedding MedDream only on the server that hosts it:

```
security.frameOptionsPolicy = SAMEORIGIN
#security.contentSecurityPolicy = frame-ancestors 'self'
```

(Results in headers `X-Frame-Options: sameorigin`, `Content-Security-Policy: frame-ancestors 'self'`.)

This disables both headers altogether, therefore MedDream can be opened in an IFRAME without any restrictions:

```
security.frameOptionsPolicy = NONE
#security.contentSecurityPolicy =
```

This disallows IFRAME embedding completely:

```
security.frameOptionsPolicy = DENY
#security.contentSecurityPolicy = frame-ancestors 'none'
```

(Results in headers `X-Frame-Options: deny`, `Content-Security-Policy: frame-ancestors 'none'`. Please note that **DICOM PDFs won’t open any more**, as MedDream uses an internal IFRAME for the system-default PDF viewer.)

This allows embedding on servers listed in the second parameter:

```
security.frameOptionsPolicy = ALLOW-FROM
security.frameOptionsWhitelist = 93.184.216.34,example.com
#security.contentSecurityPolicy = frame-ancestors 'self' 93.184.216.34␣
↪example.com
```

(Results in headers `X-Frame-Options: allow-from <list>`, `Content-Security-Policy: frame-ancestors 'self' <list>` with <list> populated from the second parameter.)

In the examples above, .contentSecurityPolicy is commented out as the same default value comes from .frameOptionsPolicy. Exotic cases might require overrides.

6. "Secure" flag of the session cookie

```
server.servlet.session.cookie.secure = true
```

After configuring `true`, MedDream must be accessed over HTTPS or else the browser won't send session cookies back to the server after a successful login. Recommended in HTTPS-only installations.

7. "SameSite" flag of the session cookie

```
server.servlet.session.cookie.sameSite = strict
```

The default value is the string "unset", which means the flag is not added to the cookie. Other possible values:

- `none` – no restrictions on the cookie, however the "Secure" flag is still mandatory;
- `lax` and `strict` will further prevent embedding MedDream in an IFRAME, and might cause problems with viewing DICOM PDFs.

8. Disable the X-XSS-Protection header if needed

```
security.xssProtectionDisabled = false
```

The header X-XSS-Protection is sent by default; `true` here will turn it off for troubleshooting etc.

9. Disable the X-Content-Type-Options header if needed

```
security.contentTypeOptionsDisable = false
```

The header X-Content-Type-Options is sent by default; `true` here will turn it off for troubleshooting etc.

10. Disable usage statistics collection

```
com.softneta.statistics.enabled = true
```

By default MedDream logs number of sessions and similar things to files statistics/YYYY-MM.stat below configRoot. As of 7.5.1+, this is done only offline, without sending collected data to Softneta.

When using the "pay per view" licensing model, these offline data provide proof for a certain amount of monthly payment. In other cases statistics can help to find the optimal number of concurrent connections or provide insights when troubleshooting performance problems.

11. Path to usage statistics files

```
com.softneta.statistics.fileLocation = ${com.softneta.meddream.configRoot}/
↪statistics
```

The default value is shown above. In dockerized installations this option might be more convenient than symlinks etc.

12. Local AE Title for saving annotations and key objects

```
com.softneta.meddream.dcmsnd.bind = MEDDREAM
```

Created DICOM objects will be sent to the PACS over DICOM C-STORE. This Local AET identifies MedDream as a sender. For testing purposes one might use any AET already accepted by the PACS.

Not used in case of the “QR” plugin because the plugin has a separate parameter for Local AET.

13. Credentials for management endpoints like /manage, /h2-console, etc

```
authentication.manager.username = LOGIN_NAME
authentication.manager.password = LOGIN_PASSWORD
```

Use a strong password, and do not leave it configured longer than necessary. (An empty username or password makes the login impossible.)

14. Enable the /manage endpoint

```
management.server.port = ${server.port}
```

This is diagnostic functionality for qualified personnel. The endpoint can be used to examine MedDream logs etc over HTTP, therefore it presents a security risk.

Do not leave it configured longer than necessary, and use a strong password in authentication.manager.password.

15. Enable the /h2-console endpoint

```
spring.h2.console.enabled = true
spring.h2.console.settings.web-allow-others = true
```

The /h2-console endpoint allows to manipulate contents of internal databases (Reporting function, auth-meddream credentials storage, job engine and so on). This is an obvious security risk.

Other places of this Manual explain the rare cases when this endpoint is needed. Generally, do not leave it configured longer than necessary, and use a strong password in authentication.manager.password.

16. Log level

The log level can be adjusted for every class or a particular part of class hierarchy. The example below provides separate parameters (not alternatives of a single parameter).

```
logging.level.ROOT = ERROR
logging.level.com.softneta = INFO
logging.level.com.softneta.meddream.dicomStoreServer = DEBUG
logging.level.com.mchange.v2.resourcepool.BasicResourcePool = WARN
```

The last example (. . . BasicResourcePool) is currently needed for the Dcm4che2 plugin, as otherwise it reports database connection problems with a too vague and misleading error message.

17. Log file name without extension

```
logging.file = meddream
```

Commenting this line out will disable logging to file. The console, if visible, will still receive the messages.

Do not include the extension; it is added automatically together with additional suffixes.

To change the location of the file, please change logging.path instead.

18. Specify a certain directory for log files

```
logging.path = /var/log
```

19. Limit size of old log files

```
logging.file.max-size = 10MB
```

The log file is automatically rotated (renamed to *.log.1, *.log.2, ...) when its size reaches this value.

20. Limit number of old log files

```
logging.file.max-history = 10
```

During log rotation, the oldest ones are removed unless this parameter is set to 0.

21. Base directory for caching thumbnails, pixel data, DICOM attributes and study structure objects

```
com.softneta.preparation.cacheDir = ./temp/cache
```

22. Directory where converted videos and conversion process logs are stored

```
com.softneta.video.convertedDir = ./temp/converted-videos
```

23. Directory to store temporary anonymized images before uploading to DICOM Library

```
com.softneta.dicom.library.upload.tempDir = ./temp/upload-to-dicom-library
```

24. Settings for Export to ISO/Burn

```
com.softneta.export.tempDir = ${com.softneta.meddream.tempDir}
com.softneta.export.tmpCleanAfterMilliSec = 86400000
com.softneta.export.additionalSoftwarePath=
#com.softneta.export.isoSizes[0].id = MiniCD
#com.softneta.export.isoSizes[0].name = Mini CD (210 MB)
#com.softneta.export.isoSizes[0].size = 220200960
com.softneta.export.appendIsoSizes = false
```

Every `tmpCleanAfterSec` seconds after application start, MedDream attempts to remove outdated (older than tmpCleanAfterSec) export jobs and their temporary files under `tempDir`. During normal operation, this data is removed after the Export window closes, provided that there are no related background downloads.

If `additionalSoftwarePath` is specified, it shall be the path to a directory with files of a DICOMDIR viewer (autorun.inf and others). They will be automatically included in each disc of exported studies.

The `isoSizes[]` array can be used to specify custom media sizes. `id` is a simple alphanumeric identifier (Latin letters, no spaces). `size` is in bytes, zero means no limit.

When `appendIsoSizes` is true, custom media sizes are appended to the default list of unlimited (0), CD (681574400), DVD (5046586572), DL-DVD (9126805504). Otherwise the default list is lost and, if some of its entries are still needed, they need to be added to `isoSizes[]` manually.

25. Possibility to disable anonymization in the Share To DICOM Library function (VET license only)

```
com.softneta.dicom.library.upload.anonymized = false
```

For non-VET license this parameter is ignored and remains set to `true`.

26. Possibility to disable the study count request for Patient History window

```
com.softneta.meddream.patientStudyCountEnabled = false
```

This is not needed under normal circumstances (value should remain `true`).

27. Bind address of the HTTP server

```
server.address = 192.168.200.100
```

This parameter can be used to expose MedDream on a single IP address that belongs to the machine. The default value is 0.0.0.0, meaning “all IP addresses”.

28. HTTP context path

```
server.servlet.context-path = /meddream
```

By default the MedDream backend is available at `/`. Some reverse proxying scenarios might benefit from different values.

29. Enable/disable compression of HTTP responses

```
server.compression.enabled = true
server.compression.mime-types = application/octet-stream
```

If the first parameter is set to true, then the second one lists comma-separated MIME types to which the compression will be applied.

30. Rename robots.txt

```
com.softneta.settings.robotsFileName = robots.txt
```

MedDream is shipped with an example file sys/settings/robots.txt. If you are experimenting with multiple files, it might be more convenient to point to a different one instead of rewriting contents of the same file.

31. “Send to DICOM Library”: upload chunk size

```
com.softneta.dicom.library.upload.partSize = 100000
```

The default is 1 MB. If the Internet connection to DICOM Library is slow, then smaller values might yield faster recovery from connection drops.

32. “Send to DICOM Library”: number of retries and related pause

```
com.softneta.dicom.library.upload.repeatTimes = 5
com.softneta.dicom.library.upload.repeatAfterSeconds = 5
```

If the upload process fails, then the related job pauses for `repeatAfterSeconds` seconds and retries up to `repeatTimes` times.

33. “Send to DICOM Library”: thread pool management

```
com.softneta.dicom.library.upload.job.corePoolSize = 10
com.softneta.dicom.library.upload.job.maxPoolSize = 100
com.softneta.dicom.library.upload.job.keepIdleForSeconds = 30
```

See https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ThreadPoolExecutor.html. Initially, `corePoolSize` threads are ready for work; up to `maxPoolSize` threads can be created on demand and they will be terminated after `keepIdleForSeconds` seconds of inactivity.

34. Local Storage SCP: AE Title

```
com.softneta.dicomStoreService.localAETitle = MEDDREAM
```

PACSes need to know this value, the corresponding IP and port for sending images to MedDream.

35. Local Storage SCP: TCP port

```
com.softneta.dicomStoreService.port = 11116
```

The SCP binds to all IP addresses that belong to the machine, and listens on this port.

36. Local Storage SCP: base directory for received DICOM files

```
com.softneta.dicomStoreService.saveURIDirectory = file:///c:/StoreSCP
```

If left unconfigured or empty, the SCP will not start.

For better compatibility with the temporary files cleaner, empty value is recommended here instead of commenting the entire setting out.

Under this directory the SCP creates received files: CALLING_AET/ STUDY_INSTANCE_UID /SERIES_INSTANCE_UID/ SOP_INSTANCE_UID.dcm. The path must be absolute and in the `file://` format. Avoid long saveURIDirectory under Windows, because together with UID values from the files the path might become longer than supported by Windows.

37. Local Storage SCP: accepted Calling AE Titles

```
com.softneta.dicomStoreService.acceptAETitles = DCMSND,SENDTOPACS
```

A comma-separated list of accepted titles. It is not possible to accept “any” title.

If left unconfigured or empty, the SCP will not start.

38. Local Storage SCP: number of simultaneous connections

```
com.softneta.dicomStoreService.multiThreadCount = 5
```

Connections beyond this number will be ignored instead of rejecting associations.

39. Default value of (0008,0005) Specific Character Set when this attribute is missing in the DICOM file

```
com.softneta.dicomParser.defaultCharset =
```

Empty value, commented out line and `US-ASCII` are equivalent.

40. Force using `defaultCharset` above even if (0008,0005) is present in the DICOM file

```
com.softneta.dicomParser.defaultCharsetOverride = false
```

`true` will ignore any declarations in DICOM files. This might be adequate in a (small) clinic where all modalities specify wrong Specific Character Set.

41. MedCAD: path to backend plugins directory

```
com.softneta.meddream.medcad.pluginsDir = ./medcadPlugins
```

The value in this example is the default.

42. MedCAD: path to frontend plugins directory

```
com.softneta.meddream.medcad.frontendPluginsDir = ./md5/plugins
```

The value in this example is the default, and currently it’s hardcoded in the frontend.

43. Background DICOM preparation (EXPERIMENTAL): thread management etc

```
com.softneta.preparation.coreThreadCount = 1
com.softneta.preparation.maxThreadCount = 40
com.softneta.preparation.workQueueSize = 2000
com.softneta.preparation.threadTimeoutSeconds = 120
com.softneta.preparation.multiFrameThreadCount = 5
com.softneta.preparation.cacheFirstSeriesInstances = 16
```

See https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ThreadPoolExecutor.html. Initially, `coreThreadCount` threads are ready for work and up to `maxThreadCount` threads can be created on demand.

Threads are terminated after `threadTimeoutSeconds` even if busy. You might need more time if some files are very large.

For every multiframe file, up to `multiFrameThreadCount` threads will be created for processing frames of the same file in parallel.

When study structure is requested from the backend, it automatically starts processing for up to cacheFirstSeriesInstances images from the beginning of every series. A lower value might decrease the system load when a study is opened. A very large value (for example, 10000) would ensure that all images are processed, and might be beneficial if the HIS automatically opens MedDream for new studies as soon as they arrive (browser-based preparation).

44. Temporary files cleaner

    The cleaner starts 30 seconds after the application start, and executes every `cleanRateMilliSec`. Files are removed by their modification date.

    You can configure multiple locations with a distinct file name pattern and maximum age. The setting `matchAnyDepth=false` disallows searching for a match in every subdirectory and beyond, thus helps to avoid needless scanning of underlying large directory trees.

    The default configuration, which is duplicated below, lists known locations with possible leftover temporary files (MedDream removes most of them ASAP during normal operation).

```
com.softneta.tempCleaner.cleanRateMilliSec=1800000

com.softneta.tempCleaner.tempItems[0].directory = ${com.softneta.meddream.
↪tempDir}/archive
com.softneta.tempCleaner.tempItems[0].pattern = *
com.softneta.tempCleaner.tempItems[0].olderThanSec = 7200

com.softneta.tempCleaner.tempItems[1].directory = ${com.softneta.meddream.
↪tempDir}
com.softneta.tempCleaner.tempItems[1].pattern = *_export
com.softneta.tempCleaner.tempItems[1].olderThanSec = 21600
com.softneta.tempCleaner.tempItems[1].matchAnyDepth = false

com.softneta.tempCleaner.tempItems[2].directory = ${com.softneta.meddream.
↪tempDir}
com.softneta.tempCleaner.tempItems[2].pattern = forward_*.dcm
com.softneta.tempCleaner.tempItems[2].olderThanSec = 7200
com.softneta.tempCleaner.tempItems[2].matchAnyDepth = false

com.softneta.tempCleaner.tempItems[3].directory = ${com.softneta.meddream.
↪tempDir}/3d
com.softneta.tempCleaner.tempItems[3].pattern = *
com.softneta.tempCleaner.tempItems[3].olderThanSec = 7200

com.softneta.tempCleaner.tempItems[4].directory = ${com.softneta.video.
↪convertedDir}
com.softneta.tempCleaner.tempItems[4].pattern = *
com.softneta.tempCleaner.tempItems[4].olderThanSec = 3600

com.softneta.tempCleaner.tempItems[5].directory = ${com.softneta.preparation.
↪cacheDir}
com.softneta.tempCleaner.tempItems[5].pattern = *
com.softneta.tempCleaner.tempItems[5].olderThanSec = 604800

com.softneta.tempCleaner.tempItems[6].directory = ${com.softneta.dicom.
↪library.upload.tempDir}
```

(continues on next page)

(continued from previous page)

```
com.softneta.tempCleaner.tempItems[6].pattern = *
com.softneta.tempCleaner.tempItems[6].olderThanSec = 86400

#com.softneta.tempCleaner.tempItems[7].directory = ${com.softneta.
↪dicomStoreService.saveURIDirectory}
#com.softneta.tempCleaner.tempItems[7].pattern = *
#com.softneta.tempCleaner.tempItems[7].olderThanSec = 86400
```

Take caution with the last entry (related to saveURIDirectory). If you uncomment it and the related `com.softneta.dicomStoreService.saveURIDirectory` setting is commented out, then the cleaner receives a literal string `${com.softneta...` and might misbehave. When not using the local DICOM receiver, you should either comment out (or remove) this entire cleaner entry, or set saveURIDirectory to an empty string (it is ignored both by receiver and cleaner).

If you use some PACS plugins that download DICOM files instead of directly accessing them on PACS file system (like "QR"), then their cache directories might need to be added here, too.

45. Google Analytics ID (when using GA to track user activity)

```
com.softneta.meddream.googleAnalyticsId = UA-000000-2
```

The GA client code is already included into MedDream, you only need to specify your website ID.

46. Lifetime of cached thumbnails, pixel data, DICOM attributes and study structure objects

```
com.softneta.ehcache.timeToIdleSeconds = 86400
com.softneta.ehcache.cacheCleanRate = 1800000
```

Every `cacheCleanRate` *milliseconds*, entries older than `timeToIdleSeconds` will be removed (please note different units).

Warning: As of 7.5.1+, the cache does not have a persistent index. If the application is restarted, any cached files become invisible to it; they won't be used any more, and won't be cleaned automatically. The *Temporary files cleaner* above provides a workaround.

5.1.4. Built-in authentication and authorization

MedDream supports a few universal authentication/authorization methods. They can be used when instructions for a particular PACS integration do not offer a dedicated (more proper) method.

Most methods are designed for the interactive login. As a result, the login form can be opened only when a particular method is chosen (for example, `spring.profiles.include = auth-pacsone`).

For authorization you can choose from the following list of permissions:

- `ADMIN` – Settings and Register,
- `SEARCH` – the Search window,
- `PATIENT_HISTORY` – Patient History,
- `UPLOAD_DICOM_LIBRARY` – Share (Send To DICOM Library window),
- `EXPORT_ISO` – the Export function related to "Download" and "Burn",
- `EXPORT_ARCH` – the Export function related to "Save",
- `FORWARD` – the Forward function,
- `3D_RENDERING` – Oblique MPR,

- REPORT_VIEW – read-only access to Reporting function,
- REPORT_UPLOAD – full access to Reporting function,
- ANONYMOUS_VIEW – on-the-fly anonymization of visible patient data (not the exported data),
- DOCUMENT_VIEW – ability to view Structured Report and PDF documents.

Warning: Due to a design bug, the VIEW_PRIVATE permission is almost always needed in authorization.defaultLoginPermissions, or else the Search window will nag with the message "Limited view access". However currently this permission is meaningful only when using the auth-pacsone method and plugins of type "PacsOne".

In a similar fashion, DOCUMENT_VIEW must be present for pre-7.5.1 functionality. Currently MedDream does not support permissions of "deny" type (existing ones are of "grant" type), therefore it is not possible to grant DOCUMENT_VIEW for all users and afterwards deny for a few ones.

Warning: REPORT_UPLOAD and REPORT_VIEW are mutually exclusive, although not all authentication methods enforce that. To avoid unexpected behavior, especially in future MedDream versions after the upgrade, do not assign both. This includes attempts to add REPORT_VIEW to all users, then "upgrade" to REPORT_UPLOAD for selected users. Until MedDream does not support deny-type permissions, it does not clean up conflicting or lesser-priority permissions.

5.1.4.1. In-memory storage

Provides both authentication (username/password) and authorization (permissions). All this data must be entered into application.properties, in plain text form.

The spring.profiles.include setting must include auth-inmemory. 7.5.2 does this by default and includes a single user named "demo", with a password known in advance that must be changed during the installation.

Elements of authentication.inmemory.users[] describe the credentials. Elements of authorization.users[] link to credentials and list the permissions. authorization.defaultLoginPermissions lists permissions common to everyone.

Example with two users:

```
spring.profiles.include = auth-inmemory
authentication.inmemory.users[0].userName = user1
authentication.inmemory.users[0].password = pass1
authentication.inmemory.users[1].userName = user2
authentication.inmemory.users[1].password = pass2
authorization.users[0].userName = user1
authorization.users[0].role = ADMIN,SEARCH,DOCUMENT_VIEW
authorization.users[1].userName = user2
authorization.users[1].role = SEARCH,ANONYMOUS_VIEW
authorization.defaultLoginPermissions = VIEW_PRIVATE,PATIENT_HISTORY,UPLOAD_DICOM_LIBRARY,
↪REPORT_UPLOAD
```

5.1.4.2. Embedded database storage

Keeps credentials and permissions in a built-in H2 database. Passwords are salted and hashed.

There is no GUI so far, management involves running SQL commands in the H2 Console.

> **Warning:** MedDream sessions are incompatible with H2 Console sessions. Use the incognito mode or another browser if MedDream must be used at the same time.
>
> Otherwise you will need to log out fully every time. The H2 Console has a "Disconnect" button. MedDream toolbar has a "Log Off" button, which is renamed to "Close" in HIS integration mode (and logs off even if the window fails to close afterwards).

After fresh installation of MedDream, enable this profile and H2 Console

```
spring.profiles.include = auth-meddream
spring.h2.console.enabled = true
spring.h2.console.settings.web-allow-others = true
authorization.defaultLoginPermissions = VIEW_PRIVATE
```

then do the following:

1. Try to log into MedDream. Make sure the login failed with "Access Denied" error and the log contains messages "Query returned no results for user . . . ", "User . . . not found".

   This first attempt will be unsuccessful as the database doesn't contain any accounts. However the file sys/databases/authdb.mv.db with all relevant tables will be automatically created for blank user and password values.

   If the logged error is different (for example, you've got a ClassCastException instead), then clear all cookies **and saved passwords** and try again.

2. Try opening :SERVER_PORT/h2-console in the browser. It will request HTTP credentials defined by authentication.manager.username and authentication.manager.password, then display the H2 Console login form. Make sure the *JDBC URL* form field contains `jdbc:h2:file:MEDDREAM_ROOT_DIR/sys/database/authdb` (with real path instead of MEDDREAM_ROOT_DIR) and both *User Name* and *Password* are empty. Then press the *Connect* button.

3. After the H2 Console opens, make sure the database contains the tables "USERS" and "PERMISSIONS", then use the following SQL expressions to create new users:

   ```
   INSERT INTO users VALUES('LOGIN_NAME', 'ENCODED_PASSWORD', 1);
   INSERT INTO permissions VALUES('LOGIN_NAME', 'PRIVILEGE_NAME');
   ```

   ENCODED_PASSWORD consists of a prefix (like `{bcrypt}`) that identifies the algorithm, and the encoded value. For the bcrypt algorithm (recommended) you can use online services like https://www.javainuse.com/onlineBcrypt (do not forget to add the prefix afterwards). Example: `INSERT INTO users VALUES('admin', '{bcrypt}$2a$10$DqbOQDCzx.inz.anVIYUoOiw.33lUiOn.PIhLWZVlIKARQEJNsjzC', 1).`

   The third column can either be `1` or `0`. Zero means a disabled account (login not possible).

   `PRIVILEGE_NAME` can be ADMIN, FORWARD, EXPORT_ISO, etc. Use multiple "INSERT INTO permissions" statements when adding more than one privilege.

4. As shown in the application.properties example, you can use authorization.defaultLoginPermissions to assign permissions that are needed for every user account.

Note: Do not forget Step 1, because then

- Step 2 will also create authdb.mv.db but without the relevant tables (unless you modify the JDBC URL in a particular cumbersome way). This state is unsuitable for continuing with Step 3;

- you can accidentally specify a non-blank username or password during Step 2, and not get any warning or error; however afterwards Core will be unable to access the database.

Provided that credentials were empty, you can retry Step 1 later which will populate the tables. Stopping Core and removing authdb.mv.db might also help.

Note: By default, application.SAMPLE.properties enables auth-inmemory and this is done on another line. There is no error message if you assign to spring.profiles.include multiple times, and the outcome might be counter-intuitive. Make sure to comment the unneeded line out.

5.1.4.3. LDAP storage

Verifies credentials against an LDAP server (use third-party tools to manage credentials and groups). Mapping between user's LDAP groups and MedDream permissions is configured in application.properties. The latter can also append permissions to individual users.

The `spring.profiles.include` setting must include `auth-openldap` or `auth-activedirectory`. There also is a variant of AD for servers with mandatory GSSAPI/Kerberos, `auth-kerberosldap`.

The main setting is `authentication.ldap.url`: hostname, port and base DN. Other settings depend on the profile (server type).

Example for OpenLDAP:

```
spring.profiles.include = auth-openldap
authentication.ldap.url = ldap://MY-SERVER:389/ou=DEPT-NAME,dc=COMPANY-NAME
authorization.defaultLoginPermissions = SEARCH,EXPORT_ARCH,EXPORT_ISO,PATIENT_HISTORY,
↪UPLOAD_DICOM_LIBRARY,DOCUMENT_VIEW
authorization.remapRoles.DOCTORS-GROUP-NAME = 3D_RENDERING
authorization.remapRoles.ADMINS-GROUP-NAME = ADMIN
authorization.remapRoles.TECHNICIANS-GROUP-NAME = FORWARD
authorization.users[0].userName = USER-NAME-1
authorization.users[0].role = ADMIN
authorization.users[1].userName = USER-NAME-2
authorization.users[1].role = FORWARD,REPORT_UPLOAD
```

To authenticate, MedDream binds to DN `uid=LOGIN-NAME,ou=People,ou=DEPT-NAME,dc=COMPANY-NAME` (the part `uid=...,ou=People` is hardcoded) using `LOGIN-NAME` and password from the login form.

Then it collects groups to which this user belongs, by searching for group objects exactly in `ou=groups,ou=DEPT-NAME,dc=COMPANY-NAME` with a filter `(memberUid=LOGIN-NAME)`; both `ou=groups` and the filter are hardcoded.

Finally, authorization.remapRoles.* and authorization.users[] are consulted for corresponding MedDream permission sets: the group `DOCTORS-GROUP-NAME` receives `3D_RENDERING`, the user `USER-NAME-1` additionally receives `ADMIN`, etc.

Example for Active Directory:

```
spring.profiles.include = auth-activedirectory
#spring.profiles.include = auth-kerberosldap
authentication.ldap.url = ldap://MY-SERVER:389/ou=DEPT-NAME,dc=COMPANY-NAME
authentication.ldap.bindUserDn = cn=MEDDREAM-APP,ou=SERVICE,ou=DEPT-NAME,dc=COMPANY-NAME
authentication.ldap.bindUserPassword = MEDDREAM-APP-PASS
authentication.ldap.userSearchBase = ou=USERS
authentication.ldap.groupSearchBase = ou=USERS
authorization.defaultLoginPermissions = SEARCH,EXPORT_ARCH,EXPORT_ISO,PATIENT_HISTORY,
↪UPLOAD_DICOM_LIBRARY,DOCUMENT_VIEW
authorization.remapRoles.DOCTORS-GROUP-NAME = 3D_RENDERING
```

(continues on next page)

(continued from previous page)

```
authorization.remapRoles.ADMINS-GROUP-NAME = ADMIN
authorization.remapRoles.TECHNICIANS-GROUP-NAME = FORWARD
authorization.users[0].userName = USER-NAME-1
authorization.users[0].role = ADMIN,REPORT_VIEW
authorization.users[1].userName = USER-NAME-2
authorization.users[1].role = FORWARD,REPORT_UPLOAD
```

To authenticate, MedDream binds to bindUserDn using bindUserPassword, searches for a user object at or below `ou=USERS` (empty userSearchBase is not allowed) with a hardcoded filter `(sAMAccountName=LOGIN-NAME)`, then binds to the found DN using the password from the login form.

Then it extracts group names from this user object. Its attributes `memberOf` (hardcoded) are expected to contain group DNs which start with `cn=GROUP-NAME,...`. Due to technical limitations, the user object found earlier is not reused – a second search is performed at or below groupSearchBase (should obviously be equal to userSearchBase) with the same hardcoded filter `(sAMAccountName=LOGIN-NAME)`.

As with OpenLDAP, authorization.remapRoles.* provide permission sets for entire groups, while authorization.users[] can add more permissions to individual users.

If your server demands GSSAPI, just use the “auth-kerberosldap” profile instead.

Note: By default, application.SAMPLE.properties enables auth-inmemory and this is done on another line. There is no error message if you assign to spring.profiles.include multiple times, and the outcome might be counter-intuitive. Make sure to comment the unneeded line out.

Warning: Spaces embedded in role names are ignored. For example, “Super Users” and “SuperUsers” will be treated as the same role.

5.1.4.4. Database connection

Generic authentication against a DBMS server (use third-party DBMS tools to manage credentials). Internal MedDream permissions are assigned to users in application.properties.

Example with two MySQL users:

```
spring.profiles.include = auth-db-connection
authentication.database.driverClassName = com.mysql.jdbc.Driver
authentication.database.jdbcUrl = jdbc:mysql://127.0.0.1:3306/pacs2
authorization.defaultLoginPermissions = VIEW_PRIVATE,EXPORT_ARCH
authorization.users[0].userName = user1
authorization.users[0].role = ADMIN,SEARCH,DOCUMENT_VIEW,REPORT_VIEW
authorization.users[1].userName = user2
authorization.users[1].role = SEARCH,REPORT_UPLOAD
```

`authorization.defaultLoginPermissions` lists permissions common to everyone.

Supported values for `authentication.database.driverClassName`:

- `com.mysql.jdbc.Driver`,
- `org.postgresql.Driver`,
- `com.microsoft.sqlserver.jdbc.SQLServerDriver`.

Note: By default, application.SAMPLE.properties enables auth-inmemory and this is done on another line. There is no error message if you assign to spring.profiles.include multiple times, and the outcome might be counter-intuitive. Make sure to comment the unneeded line out.

5.1.5. Predefined login credentials

In demo installations it is sometimes desirable to have the login form automatically filled with some working credentials.

For that it is necessary to patch md5/index.html. Find the final `</body></html>` and insert the following before it:

```
<script>
window.autoFillLogin = { username: ACCOUNT_USERNAME, password: ACCOUNT_PASSWORD }
</script>
```

For example,

```
... src="js/md.main.min.7.5.1-47d313eb039c.js"></script>
<script>
window.autoFillLogin = { username: 'demo', password: 'demo' }
</script>
</body> </html>
```

5.1.6. Reporting

Reports and attachments are saved into the same H2 database, sys/database/reportdb.*.

- The legacy option `$attachment_upload_dir` (config.php) is not implemented yet.
- Using MySQL or other “more serious” database engines instead is theoretically possible but was not tested yet.
- The structure is slightly different from the legacy “studynotes” and “attachment” tables. Access to legacy tables is not possible even after configuring MySQL.
- As of 7.5.2, there are no migration tools for legacy tables.
- Icons indicating whether a report is present (in Search and Viewer windows) are not fully implemented and always suggest creating a new report. As a result, finding studies with reports is difficult for users with the REPORT_VIEW permission.

There are two attachment size limits that you might need to configure. MedDream will inform you which specific limit was exceeded. The size of the largest file, and the total size of all files during the same upload operation, are configured separately:

```
spring.servlet.multipart.max-file-size=1MB
spring.servlet.multipart.max-request-size=10MB
```

Print templates are supported like in 7.1 MedDream and existing ones might work without modification. The file sys/report/default.html is an example with all supported value placeholders. Adjacent files CR.html, XC.html etc (named after the first modality value in the study), if present, will be used for studies with particular modalities.

Stamps in print templates are expected in form of files sys/report/stamp/LOGIN_NAME.jpg.

When Reporting is used in HIS integration mode (including integration into PacsOne web interface), MedDream currently assigns the same user identity in all saved reports. It can be configured by

```
authentication.his.username=his-user-name
```

5.1.6.1. Copying templates from one user to another

This task was sometimes needed in PHP-based MedDream and is now possible in 7.5.2, too.

In application.properties, assign non-empty credentials and uncomment spring.h2.console.*:

```
authentication.manager.username=LOGIN_NAME
authentication.manager.password=LOGIN_PASSWORD

spring.h2.console.enabled=true
spring.h2.console.settings.web-allow-others=true
```

Open /h2-console in a different browser or an "incognito" window of the same browser. The H2 Console login window appears. Specify the *JDBC URL* `jdbc:h2:file:/MD_CONFIG_ROOT/sys/database/reportdb` (example: `jdbc:h2:file:/C:/PROGRA~1/PacsOne/php/meddream/sys/database/reportdb`), *User Name* `sa`, and no password.

View the TEMPLATE table. Add the `WHERE user='SOURCE_LOGIN_NAME'` clause to the autogenerated SQL statement and confirm that the needed records exist.

Replace the SQL statement with the following and execute it:

```
insert into template (id, node, name, user, content)
select
        template_seq.nextval,
        t.node,
        t.name,
        'DESTINATION_LOGIN_NAME',
        t.content
from
        template t
where
        t.user = 'SOURCE_LOGIN_NAME'
```

In a similar fashion, view the TEMPLATE table again and confirm that DESTINATION_LOGIN_NAME now has the needed records, too. At this point a refreshed MedDream Reporting window should also display the corresponding templates.

Afterwards all four settings should be commented out for security. In general, spring.h2.console.* are also needed for editing the auth-meddream database, while authentication.manager.* are also needed for editing auth-meddream and for various investigations via /manage endpoint.

5.1.7. Java memory limits

Some example configurations included in MedDream installation archive specify 1 GB, which is sufficient for typical files.

In general, the application requires **an amount of memory no less than the size of the largest DICOM file**. Use this as a ballpark figure, and add a generous overhead for simultaneous connections. Furthermore, up to `com.softneta.preparation.maxThreadCount` files of the same study will be processed in the background simultaneously, so this should be taken into account, too, when your studies contain multiple moderately-sized files.

Experiments with a single connection show that a single 1.3 GB multiframe file opens rather slowly under a 2 GB memory limit and without problems under a 3 GB limit; consequently, 4 GB would be recommended in this case.

The memory limit is configured by the Java option `-XmxSIZE`, where SIZE can use suffixes "M", "G", etc (for example, `-Xmx1024M`).

- When you are starting the backend from the console (during *Essential configuration and the first run*, or for a quick experiment with a stopped service), then just add `-Xmx4G` etc as the first option. For example,

```
java -Xmx4G -jar MedDream-7.5.2.jar

java -Xmx4G -Djava.library.path=C:\MedDreamPACS-Premium\services\lib
-jar MedDream-7.5.2.jar
```

- When the service is already set up, then you need to adjust the application command line in the corresponding part of the installation. See the next chapter, *Running as a service*, for possible locations.

5.1.8. Enabling SSL

The single Tomcat web server port, `server.port`, can be configured for HTTPS instead of HTTP.

(Option 1) To create a self-signed certificate, you can use `keytool` from JRE:

```
keytool -genkeypair -alias tomcat -keyalg RSA -keysize 2048 -storetype PKCS12 -keystore␣
↪path/to/keystore.p12 -validity 3650 -storepass password
```

It will ask for a few identifying attributes of the certificate; it is possible to just press Enter every time. When asked if the information is correct, type `yes`. At the last prompt (regarding the key password), press Enter, too.

```
What is your first and last name?
    [Unknown]:
What is the name of your organizational unit?
    [Unknown]:
What is the name of your organization?
    [Unknown]:
What is the name of your City or Locality?
    [Unknown]:
What is the name of your State or Province?
    [Unknown]:
What is the two-letter country code for this unit?
    [Unknown]:
Is CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
[no]: yes

Enter key password for <tomcat>
    (RETURN if same as keystore password):
```

For verification, use keytool again:

```
keytool -list -v -storetype pkcs12 -keystore path/to/keystore.p12
```

(Option 2) If you already have a certificate, then conversion to a Java keystore is often still needed. The certificate must be in DER (not PEM) format.

```
keytool -import -alias tomcat -file myCertificate.crt -keystore keystore.p12 -storepass␣
↪password
```

To specify the keystore for the JAR, copy its file to configRoot and modify application.properties:

```
server.ssl.key-store-type=PKCS12
server.ssl.key-store=${com.softneta.meddream.configRoot}/keystore.p12
server.ssl.key-store-password=password
server.ssl.key-alias=tomcat
security.require-ssl=true
server.port=8443
```

`server.port` is here to minimize confusion: the current default port number, 8080, is normally used with HTTP, not HTTPS.

When not using the well-known HTTPS port 443, you might need to always add `https://` to MedDream URLs from now on.

5.1.9. The external links plugin

This plugin adds custom text-only buttons to MedDream toolbar. They execute various URLs which can contain metadata of an active image (like Study Instance UID) and are therefore suitable for additional integration with third-party systems.

The custom buttons appear below the toolbar button “Plugins” that is hidden by default. Go to Settings and search for “Plugins” in Viewer / Toolbar Settings, then choose at least “Show in desktop mode”.

To configure the plugin, modify the file medcadPlugins/links.json. By default it contains a single entry that implements a link to English version of the MedDream user manual:

```
[
  {
    "name": "Example: A link to User Manual",
    "position": "VIEWER",
    "openType": "ACTIVE_VIEWPORT_IFRAME",
    "url": "../languages/en/help"
  }
]
```

The top-level array can contain multiple entries instead of this single one. Elements of an entry:

Element	Allowed values	Explanation
name	Any string	Text on the button
position	ALL, VIEWER, SEARCH	In which parts of MedDream will the button appear
openType	NEW_WINDOW, SAME_WINDOW, ACTIVE_VIEWPORT_IFRAME	The URL can open in a new window, can replace the existing window (and entire MedDream) or just the active viewport
url	Any URL, including relative paths	Can also include `{INSTANCE}` and other placeholders from the list below
searchParams	INSTANCE, SERIES, STUDY, ACCESSION_NO, PATIENT_ID, EXAM_DATE, STORAGE	(optional) An array consisting of `"urlParamName":"INSTANCE"` etc, will be automatically added to `url` in form of `?urlParamName=dynamicValue&...`

The URL is constructed from `url` and `searchParams` by the frontend. The following dynamic metadata from the active image can be used:

- STORAGE – identifies the current PACS plugin (its `.id` in application.properties)
- INSTANCE – (0008,0018) SOP Instance UID
- SERIES – (0020,000E) Series Instance UID
- STUDY – (0020,000D) Study Instance UID
- ACCESSION_NO – (0008,0050) Accession Number
- PATIENT_ID – (0010,0020) Patient ID
- EXAM_DATE – (0008,0020) Study Date

Button examples

- Opens http://demo.example.com in a new window:

```
{
  "name": "External website demo",
  "position": "ALL",
  "openType": "NEW_WINDOW",
  "url": "http://demo.example.com/"
}
```

- Opens a specific study in the same window (for example, http://demo.example.com/?study=1.2.826.0.1.3680043.2.4852.20180403.113343677.435858665&storage=PacsOne&newParam=newParamValue):

```
{
  "name": "Metadata in URL query",
  "position": "VIEWER",
  "openType": "SAME_WINDOW",
  "url": "http://demo.example.com/",
  "searchParams": {
    "study": "STUDY",
    "storage": "STORAGE",
    "newParam": "otherValue"
  }
}
```

- Opens a study in the active viewport, the UID is specified in the path and other parameters – in the query (for example, http://demo.example.com/PacsOne/1.2.392.200036.9125.2.37100184241200.64618798380.28802?image=1.2.826.0.1.3680043.6.6054.19688.20100603165016.704.123):

```
{
  "name": "Metadata in URL query and path",
  "position": "VIEWER",
  "openType": "ACTIVE_VIEWPORT_IFRAME",
  "url": "http://demo.example.com/{STORAGE}/{STUDY}",
  "searchParams": {
    "image": "IMAGE"
  }
}
```

- Executes a custom URL handler on the client computer:

```
{
  "name": "Run a 3rd party application on the client computer",
  "position": "VIEWER",
  "openType": "SAME_WINDOW",
  "url": "applauncher:{STUDY}"
}
```

  Custom URL handlers is a Windows feature that starts a registered application with the entire URL on its command line. This example uses a generic protocol name “applauncher”. There also are success stories for Linux (using `gconftool-2`) and OSX (using a specifically crafted Application Bundle).

  MedDream includes a couple of helper files for Windows:

  – `runAppFromBrowser/runAppFromBrowser.bat` – a wrapper for 3rd-party applications. It attempts to remove the “applauncher:” prefix from the command line and passes the remainder to the application in form of `studyuid=VALUE`. You will need to adjust the path to your application, and its command line;

  – `runAppFromBrowser/runAppFromBrowserReg.bat` – registers the former file as a handler of the `applauncher` protocol. Must be run in an elevated Command Prompt window. You will need to adjust the path to runAppFromBrowser.bat and probably invent a better protocol name instead of this “applauncher”.

5.2. Running as a service

You should first make sure everything works when the Java Core is started manually. Afterwards a more standalone setup makes sense.

5.2.1. Windows

1. In the directory of the Java application, copy MedDream.NET2.exe or MedDream.NET4.exe (depends on installed .NET version) to MedDream.exe;
2. Open an elevated Command Prompt;
3. Run the command `MedDream.exe install`;
4. Start the “MedDream” service from the Windows service manager.

Note: If you need a native library from “libs.zip” mentioned earlier, then make sure it exists in the “libs” subdirectory. MedDream.xml already refers to it.

When configuring the Token Service this way, a different directory is required, preferably `/opt/token-service`. It’s because both Java applications attempt to read the same application.properties file, however the Token Service needs either a separate file or none at all.

1. In its installation directory, copy the chosen token-service.NET?.exe to token-service.exe;
2. Open an elevated Command Prompt there;
3. Run the command `token-service.exe install`;
4. Start the “MedDream Token Service” from the Windows service manager.

5.2.2. Linux (System-V init)

If your architecture is not 64-bit but 32-bit, then rename {INSTALL_DIRECTORY}/meddream-jar-wrapper.i386 to meddream-jar-wrapper.

If your system is RedHat-like (Fedora, RHEL, etc), then copy {INSTALL_DIRECTORY}/meddream.redhat to {INSTALL_DIRECTORY}/meddream. If it’s Debian-like (Ubuntu, Debian itself, etc), then copy {INSTALL_DIRECTORY}/meddream.debian to {INSTALL_DIRECTORY}/meddream.

Execute the following commands:

```
sudo ln -s {INSTALL_DIRECTORY}/meddream /etc/init.d/meddream
sudo chmod +x {INSTALL_DIRECTORY}/meddream {INSTALL_DIRECTORY}/meddream-jar-wrapper
sudo service meddream start
sudo chkconfig meddream on
```

Note: The variable BASEDIR in the script is dynamically initialized with the directory of the script itself. This means that you must leave the script near the JAR file, and place only a softlink in /etc/init.d, like in the above example. The entire script can be copied to /etc/init.d only if you adjust BASEDIR, too.

As Debian-like systems don’t have chkconfig, the last command can be replaced with `sudo update-rc.d meddream defaults`.

By changing the line `USER=root` in {INSTALL_DIRECTORY}/meddream, you can force a less privileged user that can still access DICOM files etc. However this will likely require updating the line `PIDF=/var/run/$PROG.pid`, too, as /var/run is usually a privileged location (a chown’ed subdirectory will suffice in that case).

The file {INSTALL_DIRECTORY}/meddream also contains the JRE command line, for example:

```
MD_JAR_COMMAND="java -Xmx1G -DANTLR_USE_DIRECT_CLASS_LOADING=true -Djava.library.path=./
↪lib -jar MedDream-7.5.2.jar"
```

To configure the Token Service this way, change the file names and command parameters accordingly:

- `/opt/token-service/token-service-jar-wrapper.i386` might need copying to `/opt/token-service/token-service-jar-wrapper`;
- `/opt/token-service/token-service.redhat` or `/opt/token-service/token-service.debian` is copied to `/opt/token-service/token-service`;
- in the console,

```
sudo ln -s /opt/token-service/token-service /etc/init.d/token-service
sudo chmod +x /opt/token-service/token-service /opt/token-service/token-
↪service-jar-wrapper
sudo service token-service start
sudo chkconfig token-service on
#sudo update-rc.d token-service defaults
```

- files `/opt/token-service/token-service` and `/opt/token-service/application.properties` will likely not need any changes.

5.2.3. Linux (systemd)

Create the file /etc/systemd/system/meddream.service with the following text:

```
[Unit]
Description=MedDream Viewer
After=syslog.target

[Service]
WorkingDirectory=/opt/meddream
User=root
ExecStart=/usr/bin/java -Xmx1G -DANTLR_USE_DIRECT_CLASS_LOADING=true -Djava.library.path=/
↪opt/meddream/services/lib -jar /opt/meddream/services/MedDream-7.5.2.jar
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

Note: Update `User=root` with the actual username under which the Java application will run.

Execute the following command for service autostart:

```
sudo systemctl enable meddream.service
```

To configure the Token Service this way, change the names accordingly: the file is /etc/systemd/system/token-service.service, and contains

```
[Unit]
Description=MedDream Token Service
After=syslog.target

[Service]
WorkingDirectory=/opt/token-service
User=root
ExecStart=/usr/bin/java -Xmx1G -jar /opt/token-service/token-service.jar
SuccessExitStatus=143
```

(continues on next page)

(continued from previous page)

```
[Install]
WantedBy=multi-user.target
```

The service autostart of course needs

```
sudo systemctl enable token-service.service
```

5.2.4. Linux (upstart)

Create a file /home/{user name}/.config/upstart/meddream.conf with the following text:

```
description "MedDream Viewer"
respawn
exec java -Xmx1G -DANTLR_USE_DIRECT_CLASS_LOADING=true -Djava.library.path={INSTALL_
↪DIRECTORY}/lib -jar {INSTALL_DIRECTORY}/MedDream-7.5.2.jar
```

To configure the Token Service this way, change the names accordingly: the file is /home/{user name}/.config/upstart/token-service.conf and contains

```
description "MedDream Token Service"
respawn
exec java -Xmx1G -jar /opt/token-service/token-service.jar
```

5.3. Verification checklist

Suppose you already went through the *Essential configuration and the first run* chapter, configured the authentication method and PACS plugins, set up Java applications as services, adjusted system.json and even read the *Reference for remaining configuration parameters* superficially.

This is a good opportunity for initial security tightening (see *Security considerations*, especially the “Summary” part). If during that process you accidentally disable too much of MedDream functionality, this will be noticed in subsequent steps.

Afterwards it’s time for a final functionality check-up.

- For installations with an interactive login and Search window:
  - The login form does not produce an error message and is successfully replaced by the Search window. An administrative login (with the ADMIN privilege) also succeeds and gives access to the Settings window.

    Remember that the login form is unavailable by default (results in “403 Forbidden” message) if no authentication method is configured.

    In 7.5.1+, there is no sense to have an interactive login if the Search window is disabled (the text “404” on dark background, due to features.search=false in system.json).

    Even if the integration doesn’t offer a working search functionality, you might still decide to keep the window enabled, at least for a short time – for convenient access to the Settings window via an administrative login.
  - The Search window shows studies from every configured PACS plugin (except FileSystem).
    - Results can be restricted by date interval, Patient ID, etc. Keep in mind, however, that some plugins do not support returning some fields at all, and some plugins do not support filtering by some returned fields.
    - If the plugin supports filtering by user identity, then results are different depending on who is logged in. Consult configuration of the PACS and MedDream for a suitable identity.

  - If Export or Forward are enabled (both globally and by current user's permissions), then the operation can be started from the Search window and succeeds.

- For all installations:
  - Opening a study succeeds: thumbnails are displayed instead of exclamation marks, the chosen image opens as well.

    Choose the studies wisely and cover at least different modalities.

    * Attempt to open MPEG2/MPEG4 video studies if any exist.
    * For example, if the PACS stores all images from a particular Source AE, or some historic images, at a totally different location, then you might not notice a faulty or incomplete "mappedStorageLocation" plugin setting until those images are accessed.

    When opening from the Search window, repeat the attempt for every PACS plugin that returned some search results.
  - From an open study, all functionality allowed by 1) configuration, 2) license and 3) current user's permissions, is accessible and succeeds:
    * Export (both its versions, "Burn" and "Export"),
    * Forward,
    * Share to DICOM Library,
    * Oblique MPR,
    * display of PDFs and SRs,
    * the expected set of measurements is available.

    Repeat with less-privileged users if any exist.
  - Verify that the example "demo" / "demo" user account is not present any more.
  - The Core service (and the token validation service, if used) starts successfully after rebooting the entire machine.
  - If rebranding is used, then the About window etc. looks as expected.
- For HIS integration installations:
  - In case of "unsafe" integration, check if every enabled URL parameter ("study" etc) results in opening of the right studies.
  - In case of token-based integration, also check whether MedDream correctly handles all integration options implemented in your HIS: expected set of object identifiers ("study", "accnum", etc – no less and no more), token-specified patient history, permission overrides, object restrictions (like by Patient ID), one-time tokens, limited-lifetime tokens. This will obviously involve different tokens and experiments at the HIS side.
  - If MedDream is supposed to open in an IFRAME at your webpage, then also attempt to open it from the dedicated network and using the dedicated server hostname.
- For integration into PacsOne web interface:
  - MedDream successfully opens from the list of studies (not from the list of patients, series or images).
  - Multiple studies are allowed. You should know how many of them the users might want, and configure the limit accordingly.
  - After MedDream opens this way, the expected set of features (Export, Reporting, etc) is still available to all users.

A few first days after going online can involve multiple restarts of the Core service. In 7.5.1+, each restart can leave a lot of unused temporary files that will be removed some time later. The cleanest way to verify this removal is to lower *every* "olderThanSec" setting (for example, to 1 second) temporarily and confirm that all accumulated files disappear. Nevertheless monitoring of the disk space is still advised.

5.4. Option: dockerized environment

We provide two ready-to-use Docker images that might save you some time, especially during evaluation. See

- https://hub.docker.com/r/meddream/dcm4chee-dicom-viewer for DCM4CHEE 5.x, or
- https://hub.docker.com/r/meddream/orthanc-dicom-viewer for Orthanc.

6. Additional software

MedDream can be used in tandem with several external pieces of software that are listed below.

To get the required software, please contact support@softneta.com.

6.1. 3D service

6.1.1. 3D server configuration

The XStream HDVR WebSDK runs with default system configuration (non-SSL through port **8088**) with the most common quality and performance settings. Default behavior is described below and is changed by modifying the file `foviaserver/config/configuration.xml`. The server must be restarted for any changes to take effect. The system configuration and default settings are described below:

```
<nodePort>8088</nodePort>
<useSSL>0</useSSL>
<jpegInteractive>50</jpegInteractive>
<jpegFinal>95</jpegFinal>
```

6.1.2. Network settings (ports and SSL)

`nodePort` and `useSSL` above specify the network protocol and port to be used. Separate port configuration settings are used for SSL and non-SSL, although only one can be used at one time.

The port can be changed to any valid unused port, such as 9123. After restarting your server, validate that the test app “helloFovia” still works correctly after the corresponding reconfiguration (`<FoviaServerIPaddress>.....:9123/version....`).

To enable SSL, set `useSSL` to 1 and restart your server. You can test the HTTPS protocol on the URL `https://localhost:8088/version` which uses the supplied test certificate. In a production environment, replace the supplied demo files `server.crt` and `server.key` in the directory `foviaserver/ssl` with your company’s official certificate.

6.1.3. Quality / Performance

Default values for quality and performance are explained below. Care must be taken when changing these settings since they affect all users on the system. There are two types of quality / performance metrics in the system: those that relate to network, and those that relate to the actual rendering (and ray casting).

The network quality and performance trade off affects how much bandwidth is required for transferring images over the network, and specifically the amount of JPEG compression for interactive and final quality images. This is largely independent of rendering quality that is described below.

There are two levels of JPEG compression: `jpegInteractive` is set to 50/100 and `jpegFinal` is set to 95/100. Typically during mouse interactions, many frames of quality `jpegInteractive` are generated (such as during the render() call during a mouseMove operation). These are quick rendering operations with more

aggressively compressed JPEG images. On mouseUp, a renderFinal() operation is triggered and results in a frame of quality `jpegFinal`. The rendering engine will take longer to generate higher quality images, and JPEGs with less aggressive compression will be used for these.

The server version and configuration settings are displayed at **http://localhost:8088/version**.

The 3D service creates a 3D representation of the study. In order to use the 3D service the MedDream license must have the “3D” module and the following files have to be edited.

6.1.3.1. system.json

In this file, update the parameters below `threeDimensionServiceIntegration`:

```
{
        "threeDimensionServiceIntegration": [{
                "serverIP": "127.0.0.1",
                "port": "8088",
                "protocol": "http:",
                "pathPrefix": null,
                "pathPrefixReplaceTo": null
        }]
}
```

- `serverIP`

Address of the server where the 3D service is installed. Default value: “127.0.0.1”.

- `port`

The set port of the 3D service. Default value: “8088”.

- `protocol`

The transfer protocol used by the server. Possible values: **http**, **https**.

- `pathPrefix`

Used if the 3D server is on a different machine than the PACS and the latter keeps DICOM files locally. Sets the base path of the DICOM file storage folders (like `D:\DICOM`). Default value: null.

- `pathPrefixReplaceTo`

A local equivalent of `pathPrefix`. At least this contains the path on the 3D server machine where DICOM file storage shared on the PACS machine is mounted (like `Z:\DICOM`) . Default value: null.

6.1.3.2. Apache configuration file (httpd.conf)

If outside connections to the 3D service are not allowed, then the Apache configuration needs the following lines:

```
<IfModule proxy_module>
    <IfModule proxy_http_module>
        RewriteEngine On
        RewriteCond %{REQUEST_URI}  ^/socket.io            [NC]
        RewriteCond %{QUERY_STRING} transport=websocket    [NC]
        RewriteRule /(.*)           ws://{3D_SERVICE_IP}:{3D_SERVICE_PORT}/$1 [P,L]
        ProxyPass        /socket.io http://{3D_SERVICE_IP}:{3D_SERVICE_PORT}/socket.io
        ProxyPassReverse /socket.io http://{3D_SERVICE_IP}:{3D_SERVICE_PORT}/socket.io
    </IfModule>
</IfModule>
```

Note: When SSL is used, change `ws` to `wss` and `http` to `https`.

6.2. Browser plugin

A plugin for the Chrome browser that expands a newly opened tab across several monitors. The URL is scanned for a entered keyword and if a match is found, a new window is created and expanded across the selected monitors. The plugin can be added using the Chrome extension tab.

To obtain the plugin, please contact support@softneta.com.

7. Security considerations

RATIONALE: It is an old dilemma of using “security through obscurity”. As a matter of fact, most MedDream installations tend to have a few security holes due to non-paranoid Web administrators. The very publication of this knowledge makes every old/unmaintained MedDream installation an easy target. But, we must draw a line one day so that at least new installations are secure.

7.1. Default passwords

The default application.SAMPLE.properties sets up “in-memory” authentication method with a username “demo” and password “demo”. It is included **only** to ease installation and might be known to everyone who has downloaded a recent MedDream release.

Do not forget to change it (or remove altogether) after the installation is finished.

7.2. Search engines

MedDream is shipped with the following sys/settings/robots.txt:

```
User-agent: *
Disallow: /

User-agent: AdsBot-Google
Disallow: /
```

This should prevent finding Internet-exposed MedDream login forms by searching for “MedDream” or “Softneta” (as most customers don’t use rebranding). Afterwards an attacker could check for known issues or misconfigurations.

If you still would like to index your installation, or allow indexing for a particular search engine only, then sys/settings/robots.txt can be modified accordingly (see http://www.robotstxt.org/robotstxt.html) or even renamed/removed. The backend treats the file as a static anonymous resource and simply returns contents of the file if it is found.

7.3. DCM4CHEE 2.x

Official installation instructions offer some defaults that sometimes are left unchanged.

The default database name for 2.x is “pacsdb”. You also might be tempted to call it “dcm4chee”. For a knowledgeable attacker, both provide a clear suggestion which PACS is there and what to try next.

After a mindless installation there also will be some default user accounts:

- a database user “pacs” with password “pacs”. Suitable for MedDream when using auth-db-connection;

- an internal user “admin” with password “admin” – suitable for MedDream and DCM4CHEE’s web interface when using auth-dcm4chee;
- an internal user “user” with password “user” – suitable for MedDream and DCM4CHEE’s web interface when using auth-dcm4chee.

Any of these accounts can be used to access sensitive patient data. We suggest to change passwords for all three, as soon as possible after the installation. The first one is used by DCM4CHEE to connect to the database, therefore you will also need to update the file server/default/deploy/pacs-*-ds.xml (name depends on database used).

7.4. Embedding into IFRAME etc

Availability to embed MedDream into FRAME, IFRAME or OBJECT containers allows untrusted websites to use MedDream in fraudulent manner (“clickjacking”). Since v6.2.1 this is restricted by default: only the server that hosts MedDream is allowed to wrap it with an additional container.

It is possible to adjust IFRAME permissions in the application.properties file. Valid values for `frameOptionsPolicy`: NONE, DENY, SAMEORIGIN, ALLOW-FROM. When NONE is used, then the X-Frame-Options and Content-Security-Policy headers are not sent; when ALLOW-FROM is used, then valid host must be set in parameter `security.frameOptionsWhitelist`. Another parameter, `security.contentSecurityPolicy`, can be used to explicitly control the Content-Security-Policy header. Address entries are separated using spaces.

```
security.frameOptionsPolicy=POLICY_VALUE
security.frameOptionsWhitelist=IP_ADDRESS or HOST_ADDRESS
#security.contentSecurityPolicy = frame-ancestors 'self' IP_ADDRESS HOST_ADDRESS
```

7.5. php.ini

Some PHP settings might make the installation less secure. This chapter lists most important recommendations and caveats.

Error handling

- `expose_php = Off`

  The value “Off” hides the version of PHP from webserver response headers or error messages and therefore makes it more difficult to scan for known vulnerabilities of PHP itself.

- `error_reporting = E_ALL`

  “E_ALL” is recommended for production. However during installation/troubleshooting of any server, including a production one, a more detailed choice like “E_ALL | E_STRICT” might be needed. One could have two versions of this line and comment the inactive one out.

- `display_errors = Off`
- `display_startup_errors = Off`

  Use “Off” in production for both, or else error messages visible to the end user might reveal paths to files and other sensitive information.

- `track_errors = Off`

  Can remain disabled as MedDream’s applet.php doesn’t use $php_errormsg.

- `html_errors = Off`

  “On” is not needed and, due to display_errors=Off, has no effect anyway.

- `log_errors = On`

- `error_log = /valid_path/PHP-logs/php_error.log`

  Logging to a file is preferred over *STDERR*. The latter means Apache's error_log where PHP messages will be intermixed with Apache messages and therefore harder to read.

- `ignore_repeated_errors = Off`

  "On" is OK only as a temporary measure if there are lots of identical errors, they are filling up the log and nothing else can be done about that right now.

General settings

- `doc_root = /PacsOneInstall/php`

  In 7.5+, the only part of MedDream served over the Web is applet.php when copied to PacsOne's "php" directory. In this configuration you can specify this directory or some parent part of the path.

- `open_basedir = /PacsOneInstall/php`

  Not relevant to MedDream any more.

  A non-empty setting provides more benefits (and influence) to other software, like PacsOne GUI, running on the same Apache server. Take caution and test all functionality.

- `include_path = .:/path/PHP-pear/`

  Keep this list as short as possible. This setting is not related to applet.php of MedDream, but might be used by other software if hosted on the same server.

- `extension_dir = /path/PHP-extensions/`

  Use a full path. This parameter is not relevant to MedDream any more.

- `mime_magic.magicfile = /path/PHP-magic.mime`

  Use a full path. Not relevant to MedDream any more. When using MedDream with PacsOne on the same host, PacsOne might still benefit on correct value.

- `allow_url_fopen = Off`

  "On" is not required by MedDream 7.5+.

- `allow_url_include = Off`

- `allow_webdav_methods = Off`

  "Off" for both is safer and doesn't interfere with any MedDream functionality.

- `variables_order = "GPCS"`

  A general recommendation, not relevant to MedDream any more.

- `session.gc_maxlifetime = 600`

  Not relevant to MedDream any more as sessions are now handled at Java side. Will influence behavior of PacsOne GUI, though.

  A smaller value will log the user off faster if the browser is closed, the computer is suspended or the Internet connection is cut. 10 minutes is a sane default and the legacy default is 1440 (24 minutes).

File uploads

- `file_uploads = Off`

  MedDream 7.5+ does not contain PHP-based Reporting functionality so this setting doesn't apply.

  "On" might be important for PacsOne, though, as its Study Notes and Upload Dicom Image functions work in a similar way.

- `upload_tmp_dir = /path/PHP-uploads/`

  Use a full path. A separate directory might make cleaning leftover files easier.

- `upload_max_filesize = 10M`
- `post_max_size = 40M`

  Files larger than upload_max_filesize will fail to upload. To account for overhead, ensure post_max_size > 4*upload_max_filesize and memory_limit > 16*post_max size.

- `max_file_uploads = 1`

  PacsOne always uploads a single attachment together with a Study Note so “1” is a safe choice.

Executable handling

- `enable_dl = Off`

  Not relevant to MedDream any more, and “Off” is a safer choice.

- `disable_functions = system, shell_exec, passthru, phpinfo, show_source, highlight_file, fopen_with_path, dbmopen, dbase_open, putenv, exec, popen, proc_open, move_uploaded_file, mkdir, rmdir, chmod, rename, chdir, filepro, filepro_rowcount, filepro_retrieve, posix_mkfifo`

  These functions are generally considered dangerous and applet.php of MedDream does not use them. However some might be needed for proper functioning of PacsOne. Take caution and test the configuration.

Session handling

- `session.save_path = /path/PHP-session/`

  Use an absolute path. This will ensure that both Apache-integrated and CLI versions of PHP keep session files in the same location.

- `session.name = mysessid`

  You can rename the session cookie to something less obvious.

  Not relevant to MedDream any more but other software on the same Apache server might benefit from this.

- `session.auto_start = Off`

  Not relevant to MedDream any more, and “Off” is a safer choice.

- `session.use_trans_sid = 0`
- `session.use_cookies = 1`
- `session.use_only_cookies = 1`

  Not relevant to MedDream any more, and “Off” is a safer choice. Take care with PacsOne, though.

- `session.cookie_domain = my.website.tld`

  Not relevant to MedDream any more, and “Off” is a safer choice. Take care with PacsOne, though.

- `session.use_strict_mode = 1`

  “1” or “On” is more secure. There is no effect on MedDream functionality in any case.

- `session.cookie_lifetime = 0`

  Zero is the legacy value (the session exists while the browser is not closed) and end users are accustomed to it. This is more relevant to PacsOne than MedDream.

- `session.cookie_secure = 1`

  Not relevant to MedDream any more. If Apache also hosts other software that is accessed only via HTTPS (plain HTTP is undesirable), then set this to “1”.

- `session.cookie_httponly = 1`

"1" is a general recommendation. Not relevant to MedDream any more, but might influence other software if hosted on the same server.

- `session.cache_expire = 30`

  Not relevant to MedDream any more but might influence other software if hosted on the same server.

- `session.cookie_samesite = Strict # PHP 7.3+`
- `session.sid_length = 256 # PHP 7.1+`
- `session.sid_bits_per_character = 6 # PHP 7.2+`
- `session.hash_function = 1 # PHP 7.0-7.1`
- `session.hash_bits_per_character = 6 # PHP 7.0-7.1`

  The above values are recommended and are understood by newer PHP versions.

- `session.referer_check = http://server/path`

  Not relevant to MedDream any more but might influence other software if hosted on the same server. For PacsOne, you can set here a typical URL pointing to its GUI (directories only, without home.php etc).

Less obvious settings

- `memory_limit = 50M`
- `max_execution_time = 30`

  Not relevant to MedDream any more. The setting provides more benefits (and influence) to other software running on the same Apache server.

- `report_memleaks = On`

  "On" might help to detect memory leaks earlier. Not relevant to MedDream any more.

7.6. Browser's cache

MedDream is designed with "zero footprint" concept in mind. From a security perspective this also means that no patient data is left on a client machine: after the end user logs out from MedDream, its cache does not contain any server responses with patient data.

Normally browsers obey the "do not cache" headers. Firefox, however, has a bug that allows to extract potentially sensitive data from browser's memory cache after the user logs out and doesn't close the entire browser application. On a shared computer, an attacker could later examine the memory cache and access data that isn't available to him via MedDream and HIS.

Shared computers make protecting sensitive data a lot more difficult. If this is unavoidable, then we recommend at least to avoid Firefox or teach users to also close the entire browser (not just a particular tab or one of the windows) after logout.

7.7. Firewalls

Every server, especially the ones that handle medical data, should have a working firewall.

MedDream obviously needs permission to listen at port configured by `server.port` for a typical HTTP(s) traffic. When the QueryRetrieve plugin is used with websockets (asynchronousStudyLoading=true in system.json), they operate over the same port.

Similarly, the receiver for QueryRetrieve plugin will continuously listen at `com.softneta.dicomStoreService.port` for a typical DICOM traffic.

When using Oblique MPR, every client machine will directly access the corresponding backend (threeDimensionServiceIntegration in system.json). Do not forget to allow this, too.

Outgoing network accesses might need to be whitelisted in a corporate firewall:

- the Register function downloads a license file from lic.softneta.com;
- the About window verifies availability of a new version at www.softneta.com;
- the Share function sends anonymized files to dicomlibrary.com.

7.8. Summary: Minimum relevant IT security requirements

- The operating system should be up to date with latest security patches.
- All not needed software should be removed from the system to reduce points of possible security breaches.
- Use the latest JAVA 8 version with all security patches.
- Server version should be hidden in HTTPS responses.
- Open only those firewall ports that are necessary to access MedDream (default Web port 80, and default DICOM port 11116 if used) or PACS servers.
- For HIS integration the secure token based integration is highly recommended, instead of the legacy URL integration.
- The MedDream backend should not run under a privileged user, like Linux “root” user or Windows “Administrators” user group.
- MedDream files should have limited permissions (Linux example: 0775 for files, 0664 for directories).
- The MedDream backend should always run over HTTPS (with SSL encryption). Do not use self signed certificates; if your company doesn’t have one then https://letsencrypt.org/ will provide it for free.
- Main security-related settings in application.properties:
  - `server.servlet.session.cookie.sameSite = strict`
  - `server.servlet.session.cookie.secure = true`
  - `security.xssProtectionDisabled = false`
  - `security.contentTypeOptionsDisable = false`
  - `security.contentSecurityPolicy = default-src 'self'; style-src 'self' 'unsafe-inline'; script-src 'self'; child-src 'self' blob:; connect-src 'self' *; img-src 'self' data:; worker-src 'self' blob:;`
  - `server.servlet.session.timeout` – set a reasonable duration based on network and system performance.
  - `authentication.manager.*` should be disabled. Enable it only for a short amount of time, and use a strong password (see below).
  - `spring.profiles.include = auth-inmemory`, `authentication.inmemory.*`, `authorization.users[]` should not be used in production configuration.

- MedDream should use strong passwords everywhere: at least 12 characters with uppercase, lowercase, numbers, special characters, and not listed in https://haveibeenpwned.com/Passwords.
- User roles must be only those that are absolutely necessary for a particular user. Less roles = more strict and safe account.
- `authorization.defaultLoginPermissions` should also be as minimum as possible, to prevent data breach from extended functionality abuse. User permissions should be based on user authentication and role control.
- The demo/demo user should be removed from the system.
- Search engines like Google should be disabled in robots.txt.
- Main security-related settings in system.json are the MedDream features. Disabling them also disables corresponding HTTP(s) endpoints:
  - turn off `search` if users won't use the Search window;
  - turn off `history` if users won't use the Patient History window;
  - turn off `reports` if users won't use the Reporting function;
  - remaining features can also be disabled if not needed.
- Review other MedDream settings to disable any not used functionality.

7.9. Summary: Minimum relevant IT security requirements

- The operating system should be up to date with latest security patches.
- All not needed software should be removed from the system to reduce points of possible security breaches.
- Use the latest JAVA 8 version with all security patches.
- Server version should be hidden in HTTPS responses.
- Open only those firewall ports that are necessary to access MedDream (default Web port 80, and default DICOM port 11116 if used) or PACS servers.
- For HIS integration the secure token based integration is highly recommended, instead of the legacy URL integration.
- The MedDream backend should not run under a privileged user, like Linux "root" user or Windows "Administrators" user group.
- MedDream files should have limited permissions (Linux example: 0775 for files, 0664 for directories).
- The MedDream backend should always run over HTTPS (with SSL encryption). Do not use self signed certificates; if your company doesn't have one then https://letsencrypt.org/ will provide it for free.
- Main security-related settings in application.properties:
  - `server.servlet.session.cookie.sameSite = strict`
  - `server.servlet.session.cookie.secure = true`
  - `security.xssProtectionDisabled = false`
  - `security.contentTypeOptionsDisable = false`
  - `security.contentSecurityPolicy = default-src 'self'; style-src 'self' 'unsafe-inline'; script-src 'self'; child-src 'self' blob:; connect-src 'self' *; img-src 'self' data:; worker-src 'self' blob:;`
  - `server.servlet.session.timeout` – set a reasonable duration based on network and system performance.

  - `authentication.manager.*` should be disabled. Enable it only for a short amount of time, and use a strong password (see below).
  - `spring.profiles.include = auth-inmemory`, `authentication.inmemory.*`, `authorization.users[]` should not be used in production configuration.
- MedDream should use strong passwords everywhere: at least 12 characters with uppercase, lowercase, numbers, special characters, and not listed in https://haveibeenpwned.com/Passwords.
- User roles must be only those that are absolutely necessary for a particular user. Less roles = more strict and safe account.
- `authorization.defaultLoginPermissions` should also be as minimum as possible, to prevent data breach from extended functionality abuse. User permissions should be based on user authentication and role control.
- The demo/demo user should be removed from the system.
- Search engines like Google should be disabled in robots.txt.
- Main security-related settings in system.json are the MedDream features. Disabling them also disables corresponding HTTP(s) endpoints:
  - turn off `search` if users won't use the Search window;
  - turn off `history` if users won't use the Patient History window;
  - turn off `reports` if users won't use the Reporting function;
  - remaining features can also be disabled if not needed.
- Review other MedDream settings to disable any not used functionality.

8. Localization

By default MedDream is available in English, Lithuanian and Russian; user documentation is only in English.

Warning: Softneta is not responsible for not verified translations made by licensed users. Please note that Medical Device certificates are not valid for not verified translations.

The default value of "languages" setting in sys/settings/system.json is just "en". You can specify any available language or their combination. For example, `:["en"]` can be changed to `:["lt","en","ru"]`.

9. Rebranding

MedDream can be rebranded and use custom branding information (e.g., logo, product name, contact details).

Important things to consider:

- By default the software license does not allow rebranding of the About dialog. Please contact info@softneta.com for an updated license that is fully compatible with rebranding.
- The license can also define software as a module and, as a result, hide the following parts of the About dialog: "Manufactured by", "Medical device class", "ID of the notified body", "FDA Cleared", the FDA and CE labels.
- Rebranded software does not check for recent versions; the version displayed in About dialog is always considered up to date.

Configuration and related files must be located in the directory `sys/settings/rebranding/`.

The main configuration file is 'rebranding_configuration.json'. It must contain a valid JSON object without comments and newlines. The file must be encoded in UTF-8.

When rebranding is enabled, most attributes must be specified in the file as they have no default values.

JSON object attributes:

- `isRebranded`

  Possible values: true, false

  Default value: false

  This attribute allows to turn on/off rebranding without changing a lot of other attributes.

  Example:

  ```
  "isRebranded":true
  ```

- `productName`

  Possible values: any text

  Replaces the text "MedDream" in the bottom left corner of the login form, the "Product" part of the About dialog, the title of the browser tab.

  Example:

  ```
  "productName":"MyViewer"
  ```

- `productVersion`

  Possible values: any text

  Replaces the version in the bottom left corner of the login form, and the "Version:" part of the About dialog.

  Example:

  ```
  "productVersion":"1.1.1"
  ```

- `copyright`

  Possible values: any text

  Replaces the text "Copyright 2020 © Softneta" in the bottom left corner of the login form, and below search results.

  Example:

  ```
  "copyright":"Copyright 2017 © Softneta"
  ```

- `companyLogoFile`

  Possible values: name (with extension) of an image file in the ./rebranding subdirectory.

  Image format: PNG, JPG

  Replaces the company logo in the About dialog.

  Size of the default image is 134 x 28. Images wider than 540 pixels do not always fit into the dialog. A too high image might move the buttons outside the screen so the dialog becomes unusable. Make sure to check your customization on various client systems (different display DPI, etc) from where MedDream will be opened.

  Example:

  ```
  "companyLogoFile":"companylogo.jpg"
  ```

  (refers to file `sys/settings/rebranding/companylogo.jpg`)

- `productLogoFile`

Possible values: name (with extension) of an image file in the ./rebranding subdirectory

Image format: PNG, JPG

Replaces the product logo in viewer's top left corner.

Size of default image is 170 x 40. In the viewer the height can be up to 52 px, anything more is clipped; if the width exceeds 215 px, the rest will overlap with thumbnails. When your picture is near these limits, it might need checking on various client systems to ensure consistent behavior.

Example:

```
"productLogoFile":"productlogo.jpg"
```

(refers to file `sys/settings/rebranding/productlogo.jpg`)

- `loginLogoFile`

  Possible values: name (with extension) of an image file in the ./rebranding subdirectory

  Image format: PNG, JPG

  Replaces the picture above the login form.

  Maximum dimensions are 270 x 50. Wider images will not be centered. Higher images will shift the form towards the bottom of the page.

  Example:

  ```
  "loginLogoFile":"loginlogo.jpg"
  ```

  (refers to file `sys/settings/rebranding/loginlogo.jpg`)

- `companyName`

  Possible values: any text, single line.

  Replaces the 1st line under "CONTACTS" in the About dialog.

- `streetAddr`

  Possible values: any text, single line.

  Replaces the 2nd line under "CONTACTS" in the About dialog.

- `postCode`, `city`

  Possible values: any text, single line.

  A combination of these (postCode + , + city) replace the 3rd line under "CONTACTS:" in the About dialog.

- `country`

  Possible values: any text, single line.

  Replaces the 4th line under "CONTACTS" in the About dialog.

- `email`

  Possible values: any text, single line.

  The 5th line under "CONTACTS" in the About dialog will display this text. A prefix "E-mail " is always added to the displayed text. A prefix "mailto:" is always added to the underlying address.

  If this line is clicked, MedDream opens the default email client that offers to compose an email to this recipient.

- `site` (optional)

  Possible values: any text that begins with "http" (you can use "https", too)

  The 6th line under "CONTACTS" in the About dialog will display this text. The same text is always added as the underlying address.

If this line is clicked, MedDream opens a new browser tab and navigates to this address.

A missing attribute will simply display an empty line without side effects.

- `defaultTheme` (optional)

  Possible values: `red` (default), `blue`, `green`

  Example:

  ```
  "defaultTheme":"blue"
  ```

A typical “all included” example of rebranding_configuration.json:

```
{
  "isRebranded":true,
  "productName":"MyViewer",
  "productVersion":"1.1.1",
  "copyright":"Copyright 2017 © Softneta",
  "companyLogoFile":"companylogo.jpg",
  "productLogoFile":"productlogo.jpg",
  "loginLogoFile":"loginlogo.jpg",
  "companyName":"company",
  "streetAddr":"address",
  "postCode":"postal code",
  "city":"city",
  "country":"country",
  "email":"test@test.com",
  "site":"http://www.web.com",
  "defaultTheme": "red"
}
```

Note: An override for the Web resource `/favicon.ico` is always expected under a name “favicon.ico” in the same directory, `sys/settings/rebranding/`. This file is used if it exists, isRebranded is set to true and the license allows rebranding.

10. Troubleshooting

10.1. Log files of the Java-based core

In the directory of the Java application, the file `meddreean.main.log` will be created.

To control the overall logging level, adjust the following line in `application.properties`:

```
logging.level.root=DEBUG
```

Alternatives to “`DEBUG`” are INFO, ERROR etc as per Log4j specification.

The existing setting `logging.level.com.softneta` refers only to internal MedDream logic and in some cases might be insufficient even at DEBUG level.

10.2. Browser download path selection

10.2.1. Google Chrome

You can choose a location on your computer where downloads should be saved by default or pick a specific destination for each download.

- On your computer, open Chrome.
- At the top right, click More and then **Settings**.
- At the bottom, click **Advanced**.
- Under the “Downloads” section, adjust your download settings:
- To change the default download location, click **Change** and select where you’d like your files to be saved.
- If you’d rather choose a specific location for each download, check the box next to “Ask where to save each file before downloading”.

If you didn’t change your default download location, then Google Chrome downloads files to the following locations:

- **Windows 10, 8, 7 and Vista:** \Users\<username>\Downloads
- **Mac:** /Users/<username>/Downloads
- **Linux:** /home/<username>/Downloads

10.2.2. Mozilla Firefox

Note: Changing the location of your downloads affects **all** downloaded files in this Web browser.

- Click the menu icon on the top right corner of the browser.
- Click **Preferences**.
- Click **General**.
- Click **Choose...** next to Save files to.

10.2.3. Microsoft Edge

- Open Microsoft Edge.
- Select **Settings and more** -> **Settings**.
- Under **Downloads**, select **Change**.
- In the dialog box, select a new location for your downloaded files.

10.2.4. Safari

To change the default download location of your Safari browser:

- Click on the **"Edit Menu"** -> **Preferences** -> **General tab**
- Locate the **"Save downloaded files to"** section, Click on **"Downloads"** > **"Other"...**
- Browse and indicate your new download location.

10.2.5. Internet Explorer 11

- click on the **gear icon** (top right corner of the window), and choose **"View Downloads"** from the menu
- click on the **Options** link in the bottom left corner
- click on the **Browse** button and pick another folder. Click **OK** once done.

10.3. Symptom: on Linux the loading pauses just after logging in, possibly with wrong webpage colors

On some Linux installations, the Java backend might pause for up to several minutes when a new concurrent connection is made. Immediately after the login, the user sees either an empty dark page with a progress bar spinning at the top, or an empty white page, or a white page with a black-and-white DEMO message.

The cause: the default configuration of OpenJDK suggests reading from /dev/random that can block due to exhausted entropy pool. Some systems are refilling the pool more slowly so this problem becomes more common.

Solution 1: add `-Djava.security.egd=file:///dev/./urandom` to the command that starts the Java backend. Refer to the *Running as a service* chapter for hints where this command might be located.

It is recommended to stop the service, then start the updated command manually and verify the effect. You might want to force entropy exhaustion so that the problem appears as soon as possible; use `dd if=/dev/random of=/dev/null bs=512 count=1` for that (on a different terminal as it will block, too).

Solution 2: the above might still not help with some OpenJDK builds, therefore you'll need to update system-wide configuration in the java.security file (traditionally it's `$JAVA_PATH/jre/lib/security/java.security`):

```
securerandom.source=file:///dev/./urandom
```

A. Appendix: Configuration changes since 7.5.1

Note: Names marked with # (for example, "#datasource.quartz") are internal properties, not to be changed by the end user. Most of them are also not offered in application.SAMPLE.properties to discourage any changes.

A.1. application.properties, generic

A.1.1. Renamed options

Old names will not work if still used. **Please update your configuration.**

`com.softneta.tempCleaner.cleanRateSec`

Changed to a close equivalent, `com.softneta.tempCleaner.cleanRateMilliSec`. The units are now milliseconds, not seconds; **do not forget to increase the value 1000 times after renaming**, or a nasty surprise might result.

`logging.file`

Changed to `logging.file.name`.

`security.user.name`, `security.user.password`

Changed to `authentication.manager.username` and `authentication.manager.password`, respectively.

`server.context-path`, `server.session.timeout`, `server.session.cookie.name`

Changed to `server.servlet.context-path`, `server.servlet.session.timeout` and `server.servlet.session.cookie.name`, respectively.

A.1.2. Options that are handled differently

`com.softneta.meddream.configRoot`

There now **is** a default value, equal to the current directory (Java's `user.dir` property). In most cases the option can remain commented out in application.properties.

If the default is unsuitable, then the value now requires `file://` (not just `file:`) which isn't part of the path. URIs will look like this:

```
com.softneta.meddream.configRoot=file:///opt/meddream
```

```
com.softneta.meddream.configRoot=file:///C:/PROGRA~1/PacsOne/php/meddream
```

`#datasource.quartz.url`

The ";INIT=..." part must be removed: it is incompatible with its new automatic equivalent.

`server.servlet.session.cookie.name`

(renamed from `server.session.cookie.name`) The default value is now "MEDDREAMSESSID", therefore the line is commented out in application.SAMPLE.properties.

A.1.3. Options with different visibility

The following were **added** to application.SAMPLE.properties but would have worked earlier as well:

`server.servlet.session.cookie.secure`, `server.servlet.session.cookie.sameSite`

They come commented out, as MedDream behavior didn't change in that regard.

`server.ssl.key-*`, `security.require-ssl`

An example of SSL setup.

A.1.4. New options or values

`authentication.his.username`

Login name of user in the HIS integration session. Currently is important only for Reporting.

`authentication.his.useSameSession`

If `true`, newly opened studies will be added to the existing insecure HIS integration session; if any viewer windows are still open, they will still work properly. The default value does not support multiple viewer windows like earlier.

`com.softneta.meddream.googleAnalyticsId`

ID of your website for the built-in Google Analytics client code.

`com.softneta.statistics.fileLocation`

A different path to usage statistics files. Not included in application.SAMPLE.properties because unlikely someone will need a non-default value.

`security.contentSecurityPolicy`

If empty, the Content-Security-Policy header gets the default value from security.frameOptions* like before. Otherwise this option provides the entire value for the header, not just the list of domains.

`security.xssProtectionDisabled`, `security.contentTypeOptionsDisable`

Possibility to disable headers "X-XSS-Protection" and "X-Content-Type-Options", respectively.

`spring.servlet.multipart.max-file-size`, `spring.servlet.multipart.max-request-size`

The default values are both 128K. Users of Reporting function may need to upload larger files.

`REPORT_VIEW` or `REPORT_UPLOAD` in `authorization.users[].role`, `authorization.defaultLoginPermissions`, `authorization.defaultHisPermissions`

As most of legacy Reporting functionality is back, these values control access to it at user level.

`accnum`, `patient`, `patient,accnum`, `file` in `authentication.his.valid-his-params`

More identifiers are supported.

`auth-kerberosldap` in `spring.profiles.include`

Equivalent of "auth-activedirectory" for servers that require GSSAPI/Kerberos.

A.2. application.properties, Conquest plugin 1.1.1

A.2.1. Renamed options

`com.softneta.meddream.pacs.configurations[].pathToConquestDir`

Changed to a close equivalent, `com.softneta.meddream.pacs.configurations[].pathToDicomIni`. **Do not forget to append "/dicom.ini"** (or whatever name your copy does have).

For better flexibility and security, the plugin now requires a full path to the file, instead of just a directory where a file with hardcoded name exists (likely together with many other files) .

A.3. (since 7.1.1) application.properties, FileSystem plugin 2.1.0

A.3.1. Options that are handled differently

`com.softneta.meddream.pacs.configurations[].type`

The correct value is now "FileSystem" instead of "FileSystemPlugin". **Please update your configuration.**

A.3.2. Options with different visibility

`com.softneta.meddream.pacs.configurations[].searchApiEnabled`, `com.softneta.meddream.pacs.configurations[].imageApiEnabled`

7.5.1 made these options not necessary in most cases.

A.4. system.json

Note: For brevity, pseudo-code with hierarchical property names will be used instead of quoting hierarchical chunks of JSON. For example, `features.export: false` means the same as

```
{
        "features": {
                "export":false
        }
}
```

A.4.1. New options

`features.reports: true`

Globally enables the Reporting function.

A.5. rebranding.json

A.5.1. Options that are handled differently

`productLogoFile`

The image is used only in viewer's top left corner, **no more used above the login form**.

`loginLogoFile`

The image takes up the entire area above the login form. The maximum dimensions have been changed, see *Rebranding*.

B. Appendix: Upgrading 7.5.* to 7.5.2 (Linux)

1. Stop the Core service (and the Token Service if it is used). The method depends on how you are running it, see *Running as a service*.
2. Back up the old MedDream folder:

```
mv /opt/meddream /opt/meddreamBKP
```

3. Extract new MedDream version to /opt/meddream/ so that the file /opt/meddream/MedDream-7.5.2.jar exists.
4. Copy the following old files to the new version:
   - meddream.lic
   - sys/settings/global.json
   - sys/settings/robots.txt
5. Copy application.SAMPLE.properties to application.properties. For every setting, copy the value from old application.properties.

   If the old file does not contain the same setting, then consult *Appendix: Configuration changes since 7.5.1* or the entire list *Reference for remaining configuration parameters*. Probably it has been renamed, or is totally new.

   Warning: You might be tempted to simply copy the old application.properties to the new version and then review it. This is not recommended:
   - A higher risk of keeping settings not supported any more, which might lead to confusion later.
   - No opportunity to encounter default values (or totally new settings) in application.SAMPLE.properties that are safer security-wise, especially if overlooked.
   - No opportunity to encounter comments/explanations that might be more comprehensive in newer versions of application.SAMPLE.properties.
6. If rebranding is used, then copy sys/settings/rebranding/* to the new version. The login form in 7.5.2 might need a larger image, see *Rebranding*.
7. Update file permissions (below example is for MedDream running as user “meddream”):

```
chown -R meddream:meddream /opt/meddream
find /opt/meddream/ -type d -exec chmod 775 {} \;
find /opt/meddream/ -type f -exec chmod 664 {} \;
chmod +x /opt/meddream/sys/ffmpeg/flv.sh
```

   DO NOT RUN MedDream AS ROOT.
8. Start the JAR so that sys/settings/system.json is created automatically. For every setting, copy the value from the old system.json. Consult the documentation (*System.json*) and list of configuration changes (*Appendix: Configuration changes since 7.5.1*).

   Warning: Unmodified system.json from older versions might be incompatible and result in a crash when starting the JAR.