{product.name} Administrator Guide

The {product.name} Team

Version 0.3.1

Table of Contents

System Requirements

Table 1. Requirements for users

Browser	Chrome or Safari

Table 2. Requirements to run the standalone version

Java Runtime Environment	version 8 or higher

Table 3. Requirements run the server version

Java Runtime Environment	version 8 or higher
Apache Tomcat	version 8.5 or higher (Servlet API 3.1.0)
MySQL Server	version 5 or higher

Installation

Prerequisites

- This guide assumes Debian 9.1 (Stretch). It may also work on Ubuntu with some modifications, but we do not test this. Instructions for other Linux distributions likely deviate significantly.
- It is further assumed that the user "www-data" already exists on the system and that it shall be used to run the application.
- All commands assume that you are logged in as the **root** user.

If you cannot log in as root but have to use `sudo` to become root, then the recommended way to do that is using the command `sudo su -`.

Install Java 8

You can install an Oracle Java 8 JDK using the following commands.

```
$ apt-get update
$ apt-get install dirmngr
$ echo "deb http://ppa.launchpad.net/webupd8team/java/ubuntu trusty main" | tee
/etc/apt/sources.list.d/webupd8team-java.list
$ echo "deb-src http://ppa.launchpad.net/webupd8team/java/ubuntu trusty main" | tee -a
/etc/apt/sources.list.d/webupd8team-java.list
$ apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys EEA14886
$ apt-get update
$ apt-get install oracle-java8-installer
$ apt-get install oracle-java8-set-default
```

Prepare database

For production use, it is highly recommended to use a MySQL database. In this section, we briefly describe how to install a MySQL server and how to prepare it for use with INCEpTION.

- Install MySQL

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

- make sure your MySQL server is configured for UTF-8. Check the following line is present in `/etc/mysql/mariadb.conf.d/50-server.cnf` (this is specific to Debian 9; on other systems the relevant file may be `/etc/mysql/my.cnf`):

  ```
  character-set-server = utf8
  collation-server     = utf8_bin
  ```

- also ensure the default settings for client connections to are UTF-8 in `/etc/mysql/mariadb.conf.d/50-server.cnf` (again Debian 9; likely in `/etc/mysql/my.cnf` on other systems)

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

- login to MySQL

```
$ mysql -u root -p
```

- create a database

```
mysql> CREATE DATABASE inception DEFAULT CHARACTER SET utf8 COLLATE utf8_bin ;
```

- create a database user called `inception` with the password `t0t4llYSecreT` which is later used by the application to access the database (instructions for `settings.properties` file below).

```
mysql> CREATE USER 'inception'@'localhost' IDENTIFIED BY 't0t4llYSecreT';
mysql> GRANT ALL PRIVILEGES ON inception.* TO 'inception'@'localhost';
mysql> FLUSH PRIVILEGES;
```

For production use, make sure you choose a different, secret, and secure password.

Setting up the application home folder

The application home folder is the place where INCEpTION's configuration file `settings.properties` resides and where the data is stored. Mind that if you are using a MySQL database server (recommended), then some data is also stored in the MySQL database. This is important when you plan to perform a backup, as both the home folder and the database content need to be included in the backup.

Now, let's go through the steps of setting up the application home folder creating a configuration file to access the previously created MySQL database.

- Create the application home folder. This is the directory where the settings files and projects (documents, annotations, etc.) are stored

```
$ mkdir /srv/inception
```

- Edit `/srv/inception/settings.properties` to define the database connection:

```
database.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
database.driver=com.mysql.jdbc.Driver
database.url=jdbc:mysql://localhost:3306/inception?useSSL=false
database.username=inception
database.password=t0t4llYSecreT
```

- Fix permissions of the application home folder

```
$ chown -R www-data /srv/inception
```

Running via embedded Tomcat

The INCEpTION standalone JAR with an embedded Tomcat server and can be easily set up as a UNIX service. This is the recommended way of running INCEpTION on a server.

Installing as a service

To set it up as a service, you can do the following steps. For the following example, I assume that you install the application in `/srv/inception`:

- Copy the standalone JAR file inception-app-standalone-{revnumber}.jar to `/srv/inception/inception.jar`. Note the change of the filename to `inception.jar`.
- Create the file `/srv/inception/inception.conf` with the following content

```
JAVA_OPTS="-Djava.awt.headless=true -inception.home=/srv/inception"
```

- In the previous step, you have already created the `/srv/inception/settings.properties` file. You **may optionally** configure the Tomcat port using the following line

```
server.port=18080
```

  If you need to do additional configurations of the embedded Tomcat, best refer to the documentation of Spring Boot itself.

- Make sure that the file `/srv/inception/inception.conf` is owned by the root user. If this is not the case, it will be ignored and any settings made there will not have any effect. If you start the application and instead of using the MySQL database, it is using an embedded database, then you should double-check that `/srv/inception/inception.conf` is owned by the root user.

```
$ chown root:root /srv/inception/inception.conf
```

- Change the owner/group of `/srv/inception/inception.jar` to `www-data`. When the service is started, it will run with the privileges of the user that owns the JAR file, i.e. in this case the

application will run as under the `www-data` user.

```
$ chown www-data:www-data /srv/inception/inception.jar
```

- Create a symlink from `/etc/init.d` to the `/srv/inception/inception.jar`:

```
$ ln -s /srv/inception/inception.jar /etc/init.d/inception
```

- Start INCEpTION using

```
$ service inception start
```

- Stop INCEpTION using

```
$ service inception stop
```

Running the standalone behind HTTPD

These are **optional** instructions if you want to run the application behind an Apache web-server instead of accessing it directly. This assumes that you already have the following packages installed:

- Apache Web Server
- mod_proxy
- mod_proxy_ajp
- Add the following lines to `/srv/inception/settings.properties`:

```
tomcat.ajp.port=18009
server.contextPath=/inception
server.use-forward-headers=true
```

- Edit `/etc/apache2/conf.d/inception.local.conf`

```
ProxyPreserveHost On

<Proxy ajp://localhost/inception >
  Order Deny,Allow
  Deny from none
  Allow from all
</Proxy>

<Location /inception >
  ProxyPass ajp://localhost:18009/inception timeout=1200
  ProxyPassReverse http://localhost/inception
</Location>
```

- Restart Apache web server

```
$ service apache2 restart
```

Database

INCEpTION uses an SQL database to store project and user data.

We test MySQL using a MySQL server. INCEpTION uses by default an embedded HSQLDB database. However, we recommend using the embedded database only for testing purposes. For production use, we recommend using a MySQL server. The reason for this is, that:

- we do more testing on the MySQL server and
- in the past, we had cases where we described in-place upgrade procedures that required performing SQL commands to change the data model as part of the upgrade. We promise to try avoiding this in the future. However, in case we offer advice on fixing anything directly in the database, this advice will refer to a MySQL database.

We try to keep the data model simple, so there should be no significant requirements to the database being used. Theoretically, it should be possible to use any JDBC-compatible database after adding a corresponding driver to the classpath and configuring INCEpTION to use the driver in the `settings.properties` file.

If you plan to use UTF-8 encoding for project name and tagset/tag name, make sure either of the following settings for MySQL databse a) in the `settings.properties` file, make sure that `database.url` includes

```
useUnicode=true&characterEncoding=UTF-8
```

b) change the `my.conf` MySQL databse configuration file to include the following line

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

Using HSQLDB in production

INCEpTION displays a warning in the user interface when an embedded database is being used. In case that you really want to run INCEpTION with an embedded database in production, you probably want to disable this warning. To do so, please add the following entry to the `settings.properties` file:

```
warnings.embeddedDatabase=false
```

Upgrade

Exporting/importing

- Log in to the application and export all the projects that you wish to migrate using the **Export** pane in the project settings
- Move your INCEpTION home folder to a safe location and create a new home folder in the old location
- Copy the **settings.properties** and **formats.properties** (if present) back from your moved folder
- Start the new INCEpTION version to initialize the database
- Recreate the users
  - If you are using MySQL
    - create a new database and update the **settings.properties** accordingly
    - use mysqldump to dump the tables `users` and `authorities` from the old database and load it back into the new database
  - If you are not using MySQL, you have to recreate the users manually
- Restart INCEpTION and import the previously exported projects

In-place update

This method should work when updating only a bugfix version, e.g. from 2.0.9 to 2.0.10. When performing a minor or major update, better use the exporting/importing method above.

Backup your data

- Make a copy of your INCEpTION home folder
- If you are using MySQL, make a backup of your INCEpTION database, e.g. using the mysqldump command.

Standalone service

- Stop the INCEpTION service
- Replace the `inception.jar` file with the new version
- Ensure that the file has the right owner/group (usually `www-data`)
- Start the INCEpTION service again

Separate Tomcat

- While Tomcat is running, delete the old WAR from your `webapps` folder
- Wait until Tomcat has automatically deleted the INCEpTION folder
- Stop Tomcat

- Place the new WAR file into your `webapps` folder
- Start Tomcat

Version 3.2.x to 3.3.0

- When upgrading from 3.2.x or earlier to 3.3.0 or later, Automation projects break.

Version 2.3.1 to 3.0.0

- The access permissions of the super admin have changed. Super admins can no longer access annotation, curation, and monitoring pages for all projects. They can only access them if they are annotators, admins, or curators in the respective projects. However, they still have full access to the project settings of all projects and can simply give themselves the missing permissions. **After an upgrade to 3.0.0, all super admins who require project permissions on existing projects should assign these permissions to themselves. This also applies when importing old projects.** For new projects, the creator of the project always starts with annotator, curator, and project admin permissions. If these permissions are not required by the project creator, they should be removed after project creation.

System Properties

Setting	Description	Default	Example
inception.home	INCEpTION home folder	~/.inception	/srv/inception
javamelody.disabled	Disable JavaMelody	true	false

Settings

Setting	Description	Default	Example
auth.mode	Authentication mode	database	preauth
auth.preauth.header.principal	Principal header	remote_user	*some other header*
auth.preauth.newuser.roles	Default roles for new users (comma separated)	*<none>*	ROLE_PROJECT_CREATOR
auth.user.*<username>*.roles	Extra roles for user (comma separated)	*<none>*	ROLE_ADMIN
database.dialect	Database dialect	org.hibernate.dialect.HSQLDialect	org.hibernate.dialect.MySQL5InnoDBDialect
database.driver	Database driver	org.hsqldb.jdbc.JDBCDriver	com.mysql.jdbc.Driver
database.url	JDBC connection string	*location in the application home folder*	jdbc:mysql://localhost:3306/inception?useUnicode=true&characterEncoding=UTF-8
database.username	Database username	sa	user
database.password	Database password	*unset*	pass
database.initial-pool-size	Initial database connection pool size	4	
database.min-pool-size	Minimum database connection pool size	4	
database.max-pool-size	Maximum database connection pool size	10	
backup.interval	Time between backups (seconds)	0	300 *(60 * 5 = 5 minutes)*
backup.keep.number	Maximum number of backups to keep	0	5
backup.keep.time	Maximum age of backups to keep (seconds)	0	2592000 *(60 * 60 * 24 * 30 = 30 days)*
ui.brat.sentences.number	The number of sentences to display per page	5	
style.logo	Logo image displayed in the upper-right corner	*unset*	*path to an image file*
style.header.icon…	Icons/links to display in the page header. For details, see below.	*unset*	

Setting	Description	Default	Example
warnings.embeddedDatabase	Warn about using an embedded database	true	false
warnings.unsupportedBrowser	Warn about unsupported browser	true	false
debug.showExceptionPage	Show a page with a stack trace instead of an "Internal error" page. Do not use in production!	false	true
login.message	Custom message to appear on the login page, such as project web-site, annotation guideline link, … The message can be an HTML content.	*unset*	`<span style="color:red; font-size: 200%;">Use are your own risk.</span>`
user.profile.accessible	Whether regular users can access their own profile to change their password and other profile information. This setting has no effect if the application is running in pre-authentication mode.	false	true

Configure database via environment variables

The database connection details can also be configured via environment variables. When these environment variables are present, they are preferred over the `settings.properties` file. The following environment variables can be used:

Setting	Description	Default	Example
`INCEPTION_DB_DIALECT`	Database dialect	org.hibernate.dialect.HSQLDialect	org.hibernate.dialect.MySQL5InnoDBDialect
`INCEPTION_DB_DRIVER`	Database driver	org.hsqldb.jdbc.JDBCDriver	com.mysql.jdbc.Driver
`INCEPTION_DB_URL`	JDBC connection string	*location in the application home folder*	jdbc:mysql://localhost:3306/inception?useUnicode=true&characterEncoding=UTF-8
`INCEPTION_DB_USERNAME`	Database username	sa	user
`INCEPTION_DB_PASSWORD`	Database password	*unset*	pass

Custom header icons

It is possible to add custom icons to the page header. You can declare such custom icons in the application settings file as shown in the example below. Each declaration begins with the prefix `style.header.icon.` followed by an identifier (here `myOrganization` and `mySupport`). The suffixes `.linkUrl` and `.imageUrl` indicate the URL of the target page and of the icon image respectively. Images are automatically resized via CSS. However, to keep loading times low, you should point to a reasonably small image.

The order of the icons is controlled by the ID, not by the order in the configuration file!

Example: Custom header icon

```
style.header.icon.myOrganization.linkUrl=http://my.org
style.header.icon.myOrganization.imageUrl=http://my.org/logo.png
style.header.icon.mySupport.linkUrl=http://my.org/support
style.header.icon.mySupport.imageUrl=http://my.org/help.png
```

Internal backups

Annotations are internally stored as files. Whenever a user performs an action on a document, the file is updated. It is possible to keep automatic internal backups of these files, e.g. to safeguard against crashes or bugs.

The internal backups are controlled through three properties:

Setting	Description	Default
backup.interval	Time between backups (seconds)	`0` *(disabled)*
backup.keep.number	Maximum number of backups to keep	`0` *(unlimited)*
backup.keep.time	Maximum age of backups to keep (seconds)	`0` *(unlimited)*

By default, backups are disabled (**backup.interval** is set to `0`). Changing this properties to any positive number enables internal backups. The interval controls the minimum time between changes to a document that needs to have elapsed in order for a new backup to be created.

When backups are enabled, either or both of the properties **backup.keep.number** and **backup.keep.time** should be changed as well, because their default values will cause the backups to be stored indefinitely and they will eventually fill up the disk.

The properties **backup.keep.number** and **backup.keep.time** control how long backups are keep and the maximal number of backups to keep. These settings are effective simultaneously.

Example: Make backups every 5 minutes and keep 10 backups irrespective of age

```
backup.interval    = 300
backup.keep.number = 10
backup.keep.time   = 0
```

*Example: Make backups every 5 minutes and all not older than 7 days (60 * 60 * 24 * 7 seconds)*

```
backup.interval    = 300
backup.keep.number = 0
backup.keep.time   = 604800
```

Example: Make backups every 5 minutes and keep at most 10 backups that are not older than 7 days

```
backup.interval    = 300
backup.keep.number = 10
backup.keep.time   = 604800
```

External pre-authentication

It is possible to use the application with header-based external per-authentication. In this mode, the application looks for a special HTTP header (by default `remote_user`) and if that header exists, it is taken for granted that this user has been authenticated. It will check its internal database if a user by the given name exists, otherwise it will create the user.

Pre-authentication can be enabled by setting the property `auth.mode` to `preauth`. When enabling pre-authentication mode, the default roles for new users can be controlled using the `auth.preauth.newuser.roles` property. The `ROLE_USER` is always added, even if not specified explicitly. Adding also the role `ROLE_PROEJCT_CREATOR` allows all auto-created users also to create their own projects.

Since the default administrator user is not created in pre-authentication, it is useful to also declare at least one user as an administrator. This is done through the property `auth.user.<username>.roles` where `<username>` must be replaced with the name of the user. The example below shows how the user **Franz** is given administrator permissions.

Example: Authenticate using the `remote_user` header, new users can create projects, user **Franz** *is always admin.*

```
auth.mode                     = preauth
auth.preauth.header.principal = remote_user
auth.preauth.newuser.roles    = ROLE_PROJECT_CREATOR
auth.user.Franz.roles         = ROLE_ADMIN
```

The roles specified through `auth.preauth.newuser.roles` are saved in the database when a user logs in for the first time and can be changed after creation through the user interface.

The roles added through `auth.user.<username>.roles` properties are **not** saved in the database and **cannot** be edited through the user interface.