WebAnno Administrator Guide

The WebAnno Team

Version 3.4.4

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 a WebAnno server

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 WebAnno.
- 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 of WebAnno, 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 WebAnno.

- 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 webanno DEFAULT CHARACTER SET utf8 COLLATE utf8_bin ;
```

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

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

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

Setting up the WebAnno home folder

The WebAnno home folder is the place where WebAnno's configuration file `settings.properties` resides and where WebAnno stores its data. Mind that if you are using a MySQL database server (recommended), then WebAnno also stores some data 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 a home folder for WebAnno and creating a configuration file instructing WebAnno to access the previously prepared MySQL database.

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

```
$ mkdir /srv/webanno
```

- Edit `/srv/webanno/settings.properties` to define the database connection as well as internal backup properties:

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

# 60 * 60 * 24 * 30 = 30 days
backup.keep.time=2592000

# 60 * 5 = 5 minutes
backup.interval=300

backup.keep.number=10
```

- Fix permissions in WebAnno home folder

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

Running via embedded Tomcat

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

Installing WebAnno 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 WebAnno in `/srv/webanno`:

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

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

- In the previous step, you have already created the `/srv/webanno/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/webanno/webanno.conf` is owned by the root user. If this is not the case, WebAnno will ignore it and any settings made there will not have any effect. If you start

WebAnno and intead of using the MySQL database, it is using an embedded database, then you should double-check that `/srv/webanno/webanno.conf` is owned by the root user.

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

- Change the owner/group of `/srv/webanno/webanno.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 WebAnno will run as under the `www-data` user. **Do NOT run WebAnno as root.**

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

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

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

- Enable the WebAnno service using

```
$ systemctl enable webanno
```

- Start WebAnno using

```
$ service webanno start
```

- Stop WebAnno using

```
$ service webanno stop
```

Running the standalone behind HTTPD

These are **optional** instructions if you want to run WebAnno 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/webanno/settings.properties`:

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

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

```
ProxyPreserveHost On

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

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

- Restart Apache web server

```
$ service apache2 restart
```

Running via separate Tomcat

Instead of using the WebAnno standalone JAR (recommended) which comes with an embedded Tomcat, you can also deploy the WebAnno WAR archive into a separately installed Tomcat instance.

Installing Tomcat

- Install package to install user-instances of Tomcat.

```
$ apt-get install tomcat8-user authbind
```

- Create new instance

```
$ cd /opt
$ tomcat8-instance-create -p 18080 -c 18005 webanno
$ chown -R www-data /opt/webanno
```

If WebAnno is the only application you install on your server, then you can also have WebAnno running on port 80 or port 443. In that case, substitute all instances of port `18080` in these guidelines with the respective port. Mind that running via SSL on port 443 requires additional steps that we have not yet documented. Ports lower than 1024 are privileged and the WebAnno init script will automatically use a tool called **authbind** to allow WebAnno to operate on these ports as the unprivileged www-data user.

- Configure the startup script. Edit `/etc/init.d/webanno` and add the following contents or just download the file from here and place it in `/etc/init.d`.

```
#!/bin/sh

# Licensed under the Apache License, Version 2.0:
http://www.apache.org/licenses/LICENSE-2.0

# kFreeBSD do not accept scripts as interpreters, using #!/bin/sh and sourcing.
if [ true != "$INIT_D_SCRIPT_SOURCED" ] ; then
    set "$0" "$@"; INIT_D_SCRIPT_SOURCED=true . /lib/init/init-d-script
fi
### BEGIN INIT INFO
# Provides:          webanno
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: WebAnno init script
# Description:       This file should be placed in /etc/init.d. It
#                    allows starting/stopping WebAnno using the
#                    "service" command and ensures that WebAnno starts
#                    when the system is booted.
### END INIT INFO

# Author: Richard Eckart de Castilho

NAME="WebAnno"
DAEMON=none
WEBANNO_HOME="/srv/webanno"
WEBANNO_PORT="18080"
WEBANNO_USER="www-data"
CATALINA_BASE="/opt/webanno"
AUTHBIND=""
JAVA_OPTS="-Djava.awt.headless=true -Xmx750m -XX:+UseConcMarkSweepGC
-Dwebanno.home=$WEBANNO_HOME"

setup_authbind() {
  # log_action_msg "Setting up authbind configuration for $DESC on port
$WEBANNO_PORT"
  touch /etc/authbind/byport/$WEBANNO_PORT
  chmod 500 /etc/authbind/byport/$WEBANNO_PORT
  chown $WEBANNO_USER /etc/authbind/byport/$WEBANNO_PORT
  AUTHBIND="authbind --deep"
}

tomcat_pid() {
  echo `ps -fe | grep -- "-Dcatalina.base=$CATALINA_BASE" | grep -v grep | tr -s "
"|cut -d" " -f2`
}
```

```
do_start_cmd_override() {
  if [ $WEBANNO_PORT -lt 1024 ]
  then
    setup_authbind
  fi

  su - www-data -s "/bin/bash" -c "JAVA_OPTS=\"$JAVA_OPTS\" $AUTHBIND
$CATALINA_BASE/bin/startup.sh" > /dev/null 2>&1
}

do_stop_cmd_override() {
  su - www-data -s "/bin/bash" -c "$CATALINA_BASE/bin/shutdown.sh" > /dev/null 2>&1
}

do_status() {
  local pid
  pid=$(tomcat_pid)
  if [ -n "$pid" ]
  then
    log_action_msg "Status $DESC: running"
    return 0
  else
    log_action_msg "Status $DESC: stopped"
    return 1
  fi
}
```

- Make the script executable and register it to run during system start:

```
$ chmod +x /etc/init.d/webanno
$ update-rc.d webanno defaults
```

If you deploy WebAnno on a Linux machine that is short on entropy, you can significantly decrease startup time by adding `-Djava.security.egd=file:/dev/urandom` to the `JAVA_OPTS` variable in the init script.

Now we have a dedicated Apache Tomcat instance for WebAnno installed at `/opt/webanno/` that automatically starts when the system boots and that can be managed through the usual `service` commands.

Deploying the WAR file

- Place the WebAnno WAR into the Tomcat `webapps` folder:

```
$ cp webanno-webapp-3.4.4.war /opt/webanno/webapps/webanno.war
```

Mind that the copy command above renames the WAR file to `webanno.war`! This is important so that WebAnno is accessible at the URL noted later in the present guidelines.

- Start WebAnno

```
$ service webanno start
```

- Open it with your browser at `http://localhost:18080/webanno`. If you chose to run WebAnno behind the Apache web-server use `http://localhost/webanno`. The first time, it will create a username `admin` with password `admin`. Log in with this username and proceed.

Running the WAR behind Apache HTTPD

These are **optional** instructions if you want to run WebAnno 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
- Edit `/opt/webanno/conf/server.xml` and enable AJP Connector on localhost (comment in, add address, and change port

```
<Connector port="18009" protocol="AJP/1.3" redirectPort="8443" address="127.0.0.1
"/>
```

- Disable HTTP Connector (just comment it out)

```
<!--Connector port="8080" protocol="HTTP/1.1".
           connectionTimeout="20000".
           URIEncoding="UTF-8"
           redirectPort="8443" /-->
```

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

```
ProxyPreserveHost On

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

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

- Restart Apache web server

```
$ service apache2 restart
```

Database

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

We test MySQL using a MySQL server. WebAnno 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 WebAnno 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

WebAnno displays a warning in the user interface when an embedded database is being used. In case that you really want to run WebAnno 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 into WebAnno and export all the projects that you wish to migrate using the **Export** pane in the project settings
- Move your WebAnno home folder to a safe location so that WebAnno 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 WebAnno version to initialize the database
- Recreate the users
  - If you are using MySQL
    - create a new database for the new WebAnno version 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
- When upgrading to WebAnno 2.x from a pre 2.x version, remove the **format.properties** file from the WebAnno home folder
- Restart WebAnno 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 WebAnno home folder
- If you are using MySQL, make a backup of your WebAnno database, e.g. using the mysqldump command.

Standalone service

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

Separate Tomcat

- While Tomcat is running, delete the old WAR from your `webapps` folder
- Wait until Tomcat has automatically deleted the WebAnno folder
- Stop Tomcat
- Place the new WAR file into your `webapps` folder
- Start Tomcat

Upgrading Tomcat 7 to Tomcat 8

If you have been using our installation instructions to install WebAnno on Linux, you are probably running an instance of Tomcat 6. WebAnno 3.3.0 is no longer compatible with Tomcat 6 and requires at least Tomcat 8.

To upgrade your existing instance, you can try the following procedure (adapt the procedure as necessary if you have deviated from our installation instructions):

- Stop the current WebAnno Tomcat 7 instance

```
$ service webanno stop
```

- Move your old Tomcat instance out of the way

```
$ mv /opt/webanno /opt/webanno-tomcat7
```

- Install `tomcat8-user` package (this will automatically uninstall Tomcat 7)

```
$ apt-get install tomcat8-user
```

- Create new instance

```
$ cd /opt
$ tomcat8-instance-create -p 18080 -c 18005 webanno
$ chown -R www-data /opt/webanno
```

- Copy the WAR file over to the new instance

```
$ mv /opt/webanno-tomcat7/webapps/webanno.war /opt/webanno/webapps/webanno.war
```

- Stop the new WebAnno Tomcat 8 instance

```
$ service webanno start
```

If you have made additional changes to the Tomcat 7 configuration files, e.g. changed `conf/server.xml`, please make sure to redo them in the new Tomcat 8 instance.

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
webanno.home	WebAnno home folder	~/.webanno	/srv/webanno
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 WebAnno home*	jdbc:mysql://localhost:3306/weblab?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.autoScroll	Whether to scroll the annotation being edited into the center of the page	true	
ui.brat.pageSize	The number of sentences to display per page	5	
ui.brat.singleClickSelection	Whether to select annotations with a single click	false	

Setting	Description	Default	Example
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*	
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 WebAnno 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
`WEBANNO_DB_DIALECT`	Database dialect	org.hibernate.dialect.HSQLDialect	org.hibernate.dialect.MySQL5InnoDBDialect
`WEBANNO_DB_DRIVER`	Database driver	org.hsqldb.jdbc.JDBCDriver	com.mysql.jdbc.Driver

Setting	Description	Default	Example
`WEBANNO_DB_URL`	JDBC connection string	*location in WebAnno home*	jdbc:mysql://localhost:3306/weblab?useUnicode=true&characterEncoding=UTF-8
`WEBANNO_DB_USERNAME`	Database username	sa	user
`WEBANNO_DB_PASSWORD`	Database password	*unset*	pass

Custom header icons

WebAnno allows adding custom icons to the page header. You can declare such custom icons in the WebAnno 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

WebAnno stores its annotations internally in files. Whenever a user performs an action on a document, the file is updated. It is possible to configure WebAnno to keep 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

WebAnno can be used in conjunction with header-based external per-authentication. In this mode, WebAnno 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. WebAnno 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.