

Backup and restore strategies

Simple: Make backup of server app and data dir (H2 database)

```
#!/bin/bash
BUP_PATH="/backup/teedy/"
mkdir -p $BUP_PATH
rsync -lrptR /var/docs $BUP_PATH
rsync -lrptR /opt/jetty-home-11.0.15/jetty-base/webapps $BUP_PATH
cd $BUP_PATH
FILENAME=$(date +%Y-%m-%d)-teedy.tar.gz
tar -zcvf $FILENAME $BUP_PATH/var/
chown TARGETUSER:TARGETUSER $FILENAME
```

Simple: Restore backup data dir (H2 database)

```
systemctl stop jetty11.service
cd /var; rm -rf docs
mkdir -p /var/docs
cp /backup/teedy/2018-12-30-teedy.tar.gz /var/docs
cd /var/docs
tar -xvzf 2018-12-30-teedy.tar.gz
chmod 777 /var/docs
systemctl start jetty11.service
```

Simple: Backup PostgreSQL (not if you are using H2)

```
sudo -iu postgres bash -c "pg_dump YOURDATABASE > YOURDATABASE.sql" && mv
/var/lib/postgresql/YOURDATABASE.sql "$BUP_POSTGRES"/YOURDATABASE.sql
```

Simple: Restore PostgreSQL dump

The database you want to restore must exist in your psql instance! If you move your DB from one server to another you need to recreate an empty DB first. See [Teedy with PostgreSQL](#).

```
#move the database dump file to a location where postgres user can read it, for example
/var/lib/postgres/
chown postgres /var/lib/postgresql/teedy_db.sql
su - postgres

#drop old database and create a new one
psql
drop database teedy_inventory_db;
CREATE DATABASE teedy_db WITH ENCODING 'UNICODE' LC_COLLATE 'C' LC_CTYPE 'C' TEMPLATE
template0;
GRANT ALL PRIVILEGES ON DATABASE teedy_inventory_db TO teedy_inventory;

#now import the backup db dump
psql teedy_inventory_db < teedy_inventory_db.sql
```

API Use: Bash script for API export of all documents and tags with cURL

```
#!/bin/bash
BASE_URL="https://dms.yourdomain.de"
AUTH_TOKEN=$(curl -i -X POST -d username="THEUSERNAME" -d password="THEPASSWORD"
"$BASE_URL/api/user/login" -k|grep "auth_token"|cut -c24-59)
BACKUP_DIR="/backup/teedy"
TARGET_JSON_FILE=$BACKUP_DIR"/documentlist.json"
TARGET_JSON_FILE_PARSED=$BACKUP_DIR"/documentlist.txt"
TARGET_JSON_FILE_SORTED=$BACKUP_DIR"/documentlist.sorted.txt"
DIR_ZIP=$BACKUP_DIR/"ZIP"
DIR_PDF=$BACKUP_DIR/"PDF"

#create backup directory
mkdir -p "$BACKUP_DIR"
mkdir -p "$DIR_ZIP"
mkdir -p "$DIR_PDF"

#get the documents list
#curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL/api/document/list?limit=0" -k | jq . > "$TARGET_JSON_FILE"
curl --silent -X POST -H "Cookie: auth_token=$AUTH_TOKEN" "$BASE_URL/api/document/list" -d
"limit=999999" -k | jq . > "$TARGET_JSON_FILE"
```

```

#get the tags list
curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN" "$BASE_URL/api/tag/list?limit=0" -k |
jq . > "$BACKUP_DIR"/taglist.json
#read the complete list of documents
jq -c '.|{documents}|.[]|.[]|{id}+{title}+{create_date}+{tags}' "$TARGET_JSON_FILE" >
"$TARGET_JSON_FILE_PARSED"

#make sorted list which shows number of duplicates
jq -c '.|{documents}|.[]|.[]|{title}' "$TARGET_JSON_FILE" | sort | uniq -c | sort >
"$TARGET_JSON_FILE_SORTED"

COUNT=0
TOTAL=$(jq -r '.total' $TARGET_JSON_FILE)
jq -c '.|{documents}|.[]|.[]|{id}+{title}+{create_date}' "$TARGET_JSON_FILE" | while read -r
i; do
    COUNT=$((COUNT + 1))

    #parse the line and get parameters from the line
    DOC_ID=$(jq -c '.|{id}|.id' <<< $(printf '%s\n' "$i"))
    DOC_NAME=$(jq -c '.|{title}|.title' <<< $(printf '%s\n' "$i"))
    DOC_DATE=$(jq -c '.|{create_date}|.create_date' <<< $(printf '%s\n' "$i"))

    #EXPORT_FILE_NAME=$(date -d@${DOC_DATE:0:-3} +%Y-%m-%d)_${DOC_ID:1:-1}_${DOC_NAME:1:-
1}.zip
    EXPORT_FILE_NAME=$(date -d@${DOC_DATE:0:-3} +%Y-%m-%d)_${DOC_ID:1:-1}
    echo $COUNT OF $TOTAL = $EXPORT_FILE_NAME ____ ${DOC_NAME:1:-1}

    #Export ZIP
    curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL"/api/file/zip?id="${DOC_ID:1:-1}" -k -o "$DIR_ZIP"/"$EXPORT_FILE_NAME".zip

    #Export PDF
    curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL"/api/document/"${DOC_ID:1:-
1}"/pdf?margin=10\&metadata=false\&comments=true\&fitimagetopage=true -k -o
"$DIR_PDF"/"$EXPORT_FILE_NAME".pdf
done

#logout if finished

```

```
curl --silent -X POST -H "Cookie: auth_token=$AUTH_TOKEN" "$BASE_URL/api/user/logout" -k
```

API Use: Flat Hirarchy Export + File List Overview

```
#!/bin/bash

BASE_URL="https://dms.yourdomain.de"
AUTH_TOKEN=$(curl -i -X POST -d username="THEUSER" -d password="THEPASSWORD"
"$BASE_URL/api/user/login" -k|grep "auth_token"|cut -c24-59)
BACKUP_DIR="/backup/teedy"
TARGET_DOCLIST_JSON=$BACKUP_DIR"/documentlist_forfiles.json"
TARGET_FILELIST_JSON=$BACKUP_DIR"/filelist.json"

mkdir -p "$BACKUP_DIR"
rm $TARGET_DOCLIST_JSON
rm $TARGET_FILELIST_JSON

echo "Retrieving document list"
#curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL/api/document/list?limit=0" -k | jq . > "$TARGET_DOCLIST_JSON"
curl --silent -X POST -H "Cookie: auth_token=$AUTH_TOKEN" "$BASE_URL/api/document/list" -d
"limit=999999" -k | jq . > "$TARGET_JSON_FILE"

echo "Retrieving file list based on document list"
COUNT=0
jq -c '.{documents}|.[].[]|{id}+{title}+{create_date}' "$TARGET_DOCLIST_JSON" | while read -
r i; do
    COUNT=$((COUNT + 1))
    DOC_ID=$(jq -c '.{id}|.id' <<< $(printf '%s\n' $i))
    DOC_ID=${DOC_ID:1:-1}
    curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL/api/file/list?id=$DOC_ID" >> "$TARGET_FILELIST_JSON"
    #echo -e "\n" >> "TARGET_JSON_FILE"
    echo Getting $COUNT : $DOC_ID
done

echo "Dumping files into flat hirarchy"
mkdir "$BACKUP_DIR"/flat_hirarchy/
COUNT=0
jq -c '.[].[]|.{create_date}+{name}+{id}+{document_id}+{mimetype}' "$TARGET_FILELIST_JSON" |
```

```

while read -r i; do
    COUNT=$((COUNT + 1))
    DOC_ID=$(jq -c '.|{document_id}|.document_id' <<< $(printf '%s\n' "$i"))
    DOC_ID=${DOC_ID:1:-1}
    FILE_NAME=$(jq -c '.|{name}|.name' <<< $(printf '%s\n' "$i"))
    FILE_NAME=${FILE_NAME:1:-1}
    FILE_DATE=$(jq -c '.|{create_date}|.create_date' <<< $(printf '%s\n' "$i"))
    FILE_ID=$(jq -c '.|{id}|.id' <<< $(printf '%s\n' "$i"))
    FILE_ID=${FILE_ID:1:-1}
    #MIMETYPE=$(jq -c '.|{mimetype}|.mimetype' <<< $(printf '%s\n' "$i")|sed 's#/#_#g')
    #MIMETYPE=${MIMETYPE:1:-1}
    #FILE_TYPE=$(echo "$FILE_NAME"|awk -F. '{print $(NF)}')
    #FILE_TYPE=${FILE_TYPE:0:-1}
    #EXPORT_FILE_NAME=$(date -d@${FILE_DATE:0:-3} +%Y-%m-%d)_"$FILE_ID"."$FILE_ID"."$MIMETYPE"."$FILETYPE"
    EXPORT_FILE_NAME=$(date -d@${FILE_DATE:0:-3} +%Y-%m-%d)_"$FILE_ID"."$FILE_ID"."$FILE_NAME"
    echo Getting $COUNT : $EXPORT_FILE_NAME
    curl --silent -X GET -H "Cookie: auth_token=$AUTH_TOKEN"
"$BASE_URL/api/file/$FILE_ID/data" -o "$BACKUP_DIR"/flat_hirarchy/"$EXPORT_FILE_NAME"
done

#logout if finished
curl --silent -X POST -H "Cookie: auth_token=$AUTH_TOKEN" "$BASE_URL/api/user/logout" -k

```

Version #1

Erstellt: 14 Mai 2025 13:53:04 von Mario Voigt

Zuletzt aktualisiert: 14 Mai 2025 13:54:49 von Mario Voigt