

# Kapitel 1

## Einführung

Die benötigten Prozesse für lokalisierte Versionen von *Apache OpenOffice* zu verwalten, entwickelt sich zu einem umfangreichen Projekt. Dieses soll in dieser Dokumentation dargestellt werden.

Im Folgenden sollen die notwendigen Arbeitsschritte dargestellt werden. Diese werden zu verschiedenen Zeitpunkten im Prozess durchgeführt.

### 1.1 Voraussetzungen

Zunächst wird eine virtuelle Maschine benötigt, auf der die Software *Pootle* läuft. Diese Software kann von <http://docs.translatehouse.org/projects/pootle/en/stable-2.8.x> heruntergeladen werden. Die virtuelle Maschine wird auf der ASF-Infrastruktur bereitgestellt und ist im WWW unter <https://translate.apache.org/projects/> erreichbar.

Der Zugriff per SSH erfolgt mit

```
ssh <user>@translate-ec2-va.apache.org
```

Ebenso wird eine Maschine benötigt, auf der *Apache OpenOffice* gebaut wird. Dazu gibt es einen Buildbot unter <https://ci2.apache.org/#/builders/>.

Von dort wird die erstellte *en-US* zum Download auf <https://nightlies.apache.org/openoffice/install/linsnap-42x/> bereitgestellt.

## 1.2 Sprachen

Prozesse, die dazu dienen, lokalisierte Versionen von *Apache OpenOffice* bereitzustellen, werden soweit möglich in (Shell-)Skripten abgebildet und automatisiert.

Viele Skripte iterieren über eine Liste der zu bearbeitenden Sprachen. Entsprechende Listen werden in einer gesonderten (Skript-)Datei bereitgestellt und gepflegt, die von den jeweiligen Skripten eingebunden wird.

2 `<LanguageLists.sh 2>≡`

```
#!/bin/bash

# All Languages

LANGS_ALL="aa ab af am ar ast bg ca ca_XR ca_XV cs da de el en_GB es et \
eu fa fi fr gd gl he hi hr hu hy id is it ja kab kk km kn ko ky lt lv nb \
nl nn om pl ps pt pt_BR ro ru sh si sk sl so sr sv ta th tlh tr tt_RU ug \
uk vi zh_CN zh_TW"

LANGS_INCOMPLETE="af am ar ast ca_XV cs da el es eu fa gd gl he hi hr hu \
id is it kab kk km kn ko ky lv nb nn om pl ps pt pt_BR ro ru si sk sl \
so sr sv ta th tlh tt_RU ug uk vi zh_CN zh_TW"

# LANGS="ca_XR ca_CV en_GB pt_BR tt_RU zh_CN zh_TW"

# Completely translated UI

LANGS_UI="bg ca ca_XR de en_GB et fi fr hy ja lt nl tr"

# Completely translated HELP

LANGS_HELP="de en_GB fr ja nl"

# LANGS=${LANGS_ALL}

LANGS="de uk"
```

## Kapitel 2

# Build-Umgebung

Gelegentlich ist es nützlich, eine eigene Build-Umgebung aufzusetzen.

Hierzu wird das entsprechende Git-Repository geclont. Auf dem Arbeitsrechner befindet sich im Verzeichnis */home/<user>/openoffice* das geclonte Git-Repository von **Apache OpenOffice**.

Zum Bauen werden nacheinander folgende Befehle ausgeführt:

- *cd /home/<user>/openoffice*
- *cd main*
- *dmake clean*
- *autoconf*
- *../..configuration/configure.sh*
- *./bootstrap*
- *source LinuxX86-64EnvSet.sh*
- *cd instsetoo\_native*
- *build --all -P4 -- -P4*

Im Folgenden ist das von mir genutzte Configure-Skript abgebildet.

```
3  <configure.sh 3>≡  
    #!/bin/bash  
  
    LOG=${HOME}/log_$(date +%Y%m%d%H%M%S).txt  
    touch ${LOG}
```

```

#cd ${HOME}/openoffice

# git checkout A0042XMS
#GitHash=$(git log -n1 | grep commit | cut -d ' ' -f2)
#echo $GitHash >>${LOG}


cd ${HOME}/openoffice/main


# Include the Language data
. ${HOME}/configuration/langs.sh
echo ${LANGS} >>${LOG}


./configure \
--with-dmake-path=/usr/local/bin/dmake \
--with-epm-url="https://github.com/jimjag/epm/archive/v5.0.0/epm-5.0.0.tar.gz" \
--with-system-python \
--with-lang="${LANGS}" \
--with-jdk-home=/usr/lib/jvm/java-8-openjdk-amd64 \
--with-package-format="deb rpm" \
--with-vendor="Mechtilde" \
--with-build-version="aoo42x-$(date +"%Y-%m-%d %H:%M") -'uname -sm'\n ${GitHash}" \
--with-system-stdlibs \
--enable-crashdump=yes \
--enable-category-b \
--enable-beanshell \
--enable-wiki-publisher \
--enable-bundled-dictionaries \
--enable-report-builder \
--enable-opengl \
--enable-dbus \
--without-junit \
--without-stlport \

```

```
--disable-odk \
--without-fonts \
>>${LOG} 2>&1
```

```
# --with-epm=/usr/local/bin/epm
# --with-epm-url="https://github.com/jimjag/epm/archive/v5.0.0/epm-5.0.0.tar.gz"
# --enable-report-builder enables the build of the Report Builder extension
# --enable-mysql-connector enables the build of the MySQL Connector/00o extension.
#
#                               This requires access to the MySQL Connector/C (aka libmysql)
#                               to be given, too, with either the
# --with-system-mysql          Use MySQL libraries already on system, for building the MySQL
#                               Connector/00o extension. Requires MYSQLCONFIG to point to the
#                               mysql_config executable.
#                               or
# --with-libmysql-path         Use Connector/C (libmysql) installation for building the MySQL
#                               Connector/00o extension. Usage: --with-libmysql-path=<absolute
#                               path to your Connector/C installation>
# --with-system-mysql-cppconn Use MySQL C++ Connector libraries already on system

# --enable-report-builder \
# --enable-mysql-connector \
# --with-system-mysql \
# --with-libmysql-path=/usr/include/mysql++ \
# --with-system-mysql-cppconn \
# --with-java=/usr/lib64/openjdk-11 \
# epm version 4.4.2 instead 4.2 from repo
# --with-epm=/usr/local/bin/epm \
# --enable-gstreamer
```

Danach werden noch die Befehle:

```
./bootstrap >> ${LOG}  
source LinuxX86-64Env.Set.sh  
cd instsetoo_native  
build --all -P4 -- -P4
```

ausgeführt.

## Kapitel 3

# Extrahieren der SDF-Datei en-US aus dem Code

Voraussetzung für das Extrahieren dieser Datei ist eine funktionierende Build-Umgebung und ein aktueller Build. Das Bauen wird in Kapitel 2 (Seite 3) beschrieben.

*localize* ist Teil des Quellcodes von `Apache OpenOffice` und wird aus diesem gebaut werden. Im Verzeichnis *main* oder darunter wird dann der Befehl

```
localize -e -l en-US -f en-US.sdf
```

ausgeführt. Das Verzeichnis *main*/\* ist Bestandteil des *Git*-Verzeichnisses

- *-e* schaltet den Extraktionsmodus ein.
- *-l* gibt im ISO-Code die Sprache(n) an, die extrahiert werden soll(en).

Mehrere Sprachen werden durch Kommata getrennt.

- *-f* legt die Ausgabedatei fest.

Die Funktion *localize* beinhaltet auch einen *gsicheck* (Kapitel 8, Seite 37)

Die erzeugte Datei *en-US.sdf* liegt oberhalb des Verzeichnisses *main*.

Im Falle einer eigenen Buildumgebung kann *scp* verwandt werden.

### 8KAPITEL 3. EXTRAHIEREN DER SDF-DATEI EN-US AUS DEM CODE

```
<user>@<Name des Build-Servers>:~$ scp -p \  
/home/<user>/openoffice/en-US.sdf \  
<user>@translate-ec2-va.apache.org:/home/<user>/aoo/workdir/
```

Unter <https://ci2.apache.org/#/builders> gibt es ein Build-System unter Linux. Dies wird als Buildbot betrieben.

Auf dem Buildbot wird nach einem erfolgreichen Build zur Version 4.2.x auch die Datei *en-US.sdf* erstellt. Diese Datei wird abgelegt unter <https://nightlies.apache.org/openoffice/install/linsnap-42x/en-US.sdf> und kann von dort auch auf die Maschine *translate-ec2-va.apache.org* heruntergeladen werden.

Dazu wird auf dem Pootle-Server ein Cron-Job für das Herunterladen eingerichtet. Dieser wird wöchentlich ausgeführt.

```
# m h dom mon dow    command  
0 1 * * 6 /home/mechtilde/local/bin/Download.sh
```

Der Cron-Job führt folgendes Skript aus.

```
8  <Download.sh 8>≡  
#!/bin/bash  
cd ${HOME}  
wget https://nightlies.apache.org/openoffice/install/linsnap-42x/en-US.sdf .
```



## Kapitel 4

# Gemeinsame Bestandteile der Skripte

Die folgenden Skripte sind zur Verwendung auf dem Pootle-Server bestimmt.

### 4.1 Lizenz-Text

Die Skripte beginnen alle mit dem Lizenz-Vermerk. Verwendet wird die Lizenz *Apache-2.0*.

```
9  <License 9>≡ (13 26 28 33 37 39 41 43 45 47 49 50)
    #
    # License:   Apache-2.0
    # Licensed to the Apache Software Foundation (ASF) under one
    # or more contributor license agreements.
    # The ASF licenses this file to you under the Apache License,
    # Version 2.0 (the "License"); you may not use this file except
    # in compliance with the License. You may obtain a copy of the
    # License at
    #
    # http://www.apache.org/licenses/LICENSE-2.0
```

```
#
# Unless required by applicable law or agreed to in writing,
# software distributed under the License is distributed on an
# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
# KIND, either express or implied. See the License for the
# specific language governing permissions and limitations
# under the License.

# Authors: Michael Stehmann <mikeadvo@apache.org>
#          Mechtilde Stehmann <mechtilde@apache.org>
```

## 4.2 Logdatei

Die Skripte zeichnen ihren Ablauf und Fehlermeldungen in einer Log-Datei auf. Der Name der Log-Datei ist der Name des jeweiligen Skriptes mit dem Anhang *.log*.

```
10  <Logging 10>≡ (13 26 28 33 39 41 43 45 49)
    # Logging error messages
    LOG_PATH=${HOME}/log
    LOG_FILE=${LOG_PATH}/${ScriptName}"-"$(date +%Y-%m-%d-%H%M%S).log
    exec 2>>${LOG_FILE}
    echo "Messages from "$0 >>${LOG_FILE}
```

## 4.3 Pootle-Umgebung

Aktuelle Versionen von Pootle laufen in einer „virtuellen Umgebung“. Dort befindet sich auch der benötigte Python(2)-Interpreter.

Das Skript weist Variablen die benötigten Pfade in die „virtuelle Umgebung“, ins Verzeichnis mit den Übersetzungen und ins Arbeitsverzeichnis des Nutzers zu.

```
11  <PootleEnv 11>≡ (13 26 39 41 43 45 47 49 50)
    # Newer pootle versions run inside a virtual environment
    VIRTUAL_ENV=/x1/www/pootle/env
    POOTLE_SETTINGS=${VIRTUAL_ENV}/pootle.conf
    PYTHONPATH=${VIRTUAL_ENV}/lib/python2.7/site-packages
    PATH="$VIRTUAL_ENV/bin:$PATH"

    export VIRTUAL_ENV
    export PATH
    export POOTLE_SETTINGS
    export PYTHONPATH

    PROJECT_PATH=/x1/www/po

    # Directory of the pootle command and python2
    POOTLE_DIR=${VIRTUAL_ENV}/bin

    # Work directory
    WORK_DIR=${HOME}/aoo/workdir
```

## 4.4 Import des Skriptes *LanguageLists.sh*

Es wird die Datei, welche die zu bearbeitenden Sprachen enthält (s. Kapitel 1.2, Seite 2).

```
12  <Langs 12>≡ (13 26 28 33 43 45 47 49)
    # Include the Language data
    . ${HOME}/local/bin/LanguageLists.sh
    echo ${LANGS}
    echo ${LANGS} >>${LOG_FILE}
```

## Kapitel 5

# Von der SDF-Datei in die Pootle-Datenbank

Das Skript *UpdateTranslation.sh* führt alle Schritte aus, um die Pootle-Datenbank auf der Grundlage der aus dem Quellcode gewonnenen SDF-Datei zu aktualisieren. Dabei werden auch die POT- und PO-Dateien aktualisiert.

### 5.1 UpdateTranslation.sh – Kopf

Im Anfang des Skriptes sind alle gemeinsamen Bestandteile (Kapitel 4, Seite 9) enthalten.

```
13  <UpdateTranslation.sh 13>≡
    #!/bin/bash

    ScriptName="UpdateTranslation" #(.sh)

    # Version 0.1.0

    # This file should be in ~/local/bin/.

    <License 9>

    <Logging 10>

    <PootleEnv 11>

    TEMPLATES_DIR=/x1/www/po/aoo40/templates
```

*Langs* 12

*UpdateTranslation3* 14

## 5.2 Bereinigung der Arbeitsverzeinsisse

Das Arbeitsverzeichnis *\$WORK\_DIR* lautet *\$HOME/aoo/workdir*. Darin werden folgende Schritte ausgeführt:

- Prüfung auf Existenz der (aktuellen) en-US.sdf (vorsorglich)
- Entfernen der alten POT-Dateien
- Neuerstellen der Verzeichnisse für die POT-Dateien

14 *UpdateTranslation3* 14  $\equiv$  (13)

```
# If the (non-empty) file en-US.sdf is in ${HOME}:
if [ -s ${HOME}/en-US.sdf ]
then
mv ${HOME}/en-US.sdf ${WORK_DIR}/en-US.sdf
```

```
# To be carefull
rm -r ${WORK_DIR}/POT-UI/
rm -r ${WORK_DIR}/POT-HELP/helpcontent2
```

```
mkdir -p ${WORK_DIR}/POT-UI
mkdir -p ${WORK_DIR}/POT-HELP
```

*UpdateTranslation4* 15

## 5.3 Vorlagenerstellung und Import in Pootle-DB

Dazu wird derzeit auf dem Pootle-Server das in der virtuellen Umgebung installierte *Translate-Toolkit* in der Version 2.2.5 verwendet.

- *en-US.sdf* in POT-Dateien umwandeln
- Verschieben des Verzeichnisses *helpcontent2*, um die Hilfetexte (HELP) von den Anwenderschnittstellen-Texten (UI) zu trennen

```
VIRTUAL_ENV=/x1/www/pootle/env
POOTLE_DIR=${VIRTUAL_ENV}/bin
```

Die Umwandlung von SDF nach POT erfolgt mit *oo2po* aus dem Paket *translate-toolkit*. Zur Erzeugung der Verzeichnisse mit den POT-Dateien wird der Schalter *--pot* verwendet.

Auf dem Pootle-Server sind die Texte der Bedieneroberfläche und die Hilfetexte 2 Projekte. Daher müssen die Hilfetexte separiert werden.

15  $\langle \textit{UpdateTranslation4} \ 15 \rangle \equiv$  (14)

```
# Convert en-US.sdf to templates
${POOTLE_DIR}/python ${POOTLE_DIR}/oo2po --progress=verbose \
--pot ${WORK_DIR}/en-US.sdf ${WORK_DIR}/POT-UI/

# Move helpcontent2 to POT-HELP to separate HELP from UI
mv ${WORK_DIR}/POT-UI/helpcontent2 ${WORK_DIR}/POT-HELP/
```

$\langle \textit{UpdateTranslation5} \ 16 \rangle$

## 16KAPITEL 5. VON DER SDF-DATEI IN DIE POOTLE-DATENBANK

- Kopieren der Dateien ins Projektverzeichnis

```
PROJECT_PATH=/x1/www/po
```

```
WORK_DIR=${HOME}aoo/workdir
```

16     $\langle \textit{UpdateTranslation5} \ 16 \rangle \equiv$  (15)

```
# Copy PotFiles-* to project directory
```

```
cp --recursive --verbose ${WORK_DIR}/POT-UI/* \
```

```
${PROJECT_PATH}/aoo40/templates/
```

```
cp --recursive --verbose ${WORK_DIR}/POT-HELP/* \
```

```
${PROJECT_PATH}/aoo40help/templates/
```

$\langle \textit{UpdateTranslation6} \ 17 \rangle$



Für die nächsten Schritte werden Befehle aus dem Paket *Pootle* genutzt<sup>1</sup>.

```
VIRTUAL_ENV=/x1/www/pootle/env
POOTLE_DIR=${VIRTUAL_ENV}/bin
```

- Aktualisieren der Vorlagen-Einträge in die Datenbank

Nachdem neue oder aktualisierte Dateien (hier: die Vorlagen – *.pot*-Dateien) auf den Pootle-Server hochgeladen wurden, sind sie in die Datenbank einzupflegen, damit beispielsweise der aktuelle Übersetzungsstand ersichtlich wird.

Dies geschieht getrennt für die Vorlagen der Bedieneroberfläche und der Hilfe.

17  $\langle \textit{UpdateTranslation6} \ 17 \rangle \equiv$  (16)

```
# Update the translation templates into the Pootle database
${POOTLE_DIR}/python ${POOTLE_DIR}/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
update_stores --project=aoo40 --language=templates

${POOTLE_DIR}/python ${POOTLE_DIR}/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
update_stores --project=aoo40help --language=templates
```

$\langle \textit{UpdateTranslation7} \ 18 \rangle$

---

<sup>1</sup><https://pootle.translatehouse.org/>

## 5.4 Abgleich der einzelnen Sprachen mit den aktuellen Vorlagen

In den einzelnen Sprachen werden wiederum zunächst die Dateien der Benutzeroberfläche und dann der Hilfe behandelt.

Der Programmfortschritt wird in der Log-Datei dokumentiert.

Während *pootle update\_stores* den Inhalt der Datenbank an den Inhalt der *po*- bzw *pot*-Dateien anpasst, schreibt *pootle sync\_stores* die Daten aus der Datenbank in die *po*-Dateien im Dateisystem des Pootle-Servers.

18  $\langle \textit{UpdateTranslation7} \ 18 \rangle \equiv$  (17)

```
# Reconcile languages with templates
for Language in ${LANGS}
do
echo "Sync_stores UI "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POTLE_DIR}/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
sync_stores --project=aoo40 --language=${Language}
```

$\langle \textit{UpdateTranslation8} \ 19 \rangle$

#### 5.4. ABGLEICH DER EINZELNEN SPRACHEN MIT DEN AKTUELLEN VORLAGEN<sup>19</sup>

19      $\langle \textit{UpdateTranslation8} \ 19 \rangle \equiv$  (18)

```
mkdir -p ${WORK_DIR}/PO-Files/${Language}/
cd ${WORK_DIR}/PO-Files/

cp --recursive ${PROJECT_PATH}/aoo40/${Language} .
cp --recursive ${PROJECT_PATH}/aoo40/${Language} ../aoo40

cp --recursive ${PROJECT_PATH}/aoo40help/${Language}/helpcontent2 \
./${Language}
cp --recursive ${PROJECT_PATH}/aoo40help/${Language} \
../aoo40help/

# Commands for updating git repository
cd ${WORK_DIR}
git add .
git commit -am "update $(date) ${Language}"
```

$\langle \textit{UpdateTranslation9} \ 20 \rangle$

## 20 KAPITEL 5. VON DER SDF-DATEI IN DIE POOTLE-DATENBANK

*Pomerge* ist ein Werkzeug aus dem Paket *translate-toolkit*.

*Pomerge* fügt korrigierte PO-Dateien (oder -Snippets) in die bestehenden PO-Dateien ein.

*Pomerge* versucht außerdem, den Text so klein wie möglich zu ändern, damit Sie die Änderungen in Ihrem Versionskontrollsystem leichter erkennen können.

Verwendung:

```
pomerge [options] [-t <template>] -i <input> -o <output>
```

*Template* ist ein Satz von Referenzdateien, hier ein Satz von *POT*-Dateien

*input* enthält die korrigierten Dateien, die den Inhalt von *output* überschreiben.

Diese Dateien werden hier dem Arbeitsverzeichnis des Nutzers entnommen.

*output* enthält die Dateien, deren Inhalt durch *input* überschrieben werden soll.

Dies sind hier die Dateien des Projektes.

20  $\langle \textit{UpdateTranslation9} \ 20 \rangle \equiv$  (19)

```
echo "Pomerge UI "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POOTLE_DIR}/pomerge \
--template=${PROJECT_PATH}/aoo40/templates/ \
--input ${WORK_DIR}/aoo40/${Language}/ \
--output ${PROJECT_PATH}/aoo40/${Language}/
```

$\langle \textit{UpdateTranslation10} \ 21 \rangle$

#### 5.4. ABGLEICH DER EINZELNEN SPRACHEN MIT DEN AKTUELLEN VORLAGEN<sup>21</sup>

Nun wird der Inhalt der aktualisierten Dateien in die Datenbank einge-  
spielt.

Anschließend werden sie ins Arbeitsverzeichnis des Nutzers kopiert, um  
dieses zu aktualisieren.

21  $\langle \text{UpdateTranslation10 } 21 \rangle \equiv$  (20)

```
echo "Update_stores UI "${Language} >>${LOG_FILE}
```

```
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}
```

```
${POOTLE_DIR}/python ${POOTLE_DIR}/pootle \
```

```
--config ${VIRTUAL_ENV}/pootle.conf \
```

```
update_stores --project=aoo40 --language=${Language}
```

```
cp --recursive ${PROJECT_PATH}/aoo40/${Language} ${WORK_DIR}/aoo40/
```

$\langle \text{UpdateTranslation11 } 22 \rangle$

## 22KAPITEL 5. VON DER SDF-DATEI IN DIE POOTLE-DATENBANK

Die soeben beschriebenen Verfahren werden auch auf die Hilfe-Texte angewandt.

22  $\langle \text{UpdateTranslation11 } 22 \rangle \equiv$  (21)

```
echo "Sync_stores HELP "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POOTLE_DIR}/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
sync_stores --project=aoo40help --language=${Language}

echo "Pomerge HELP "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POOTLE_DIR}/pomerge \
--template=${PROJECT_PATH}/aoo40help/templates/ \
--input ${WORK_DIR}/aoo40help/${Language}/ \
--output ${PROJECT_PATH}/aoo40help/${Language}/

echo "Update_stores HELP "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POOTLE_DIR}/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
update_stores --project=aoo40help --language=${Language}

cp --recursive ${PROJECT_PATH}/aoo40help/${Language}/helpcontent2 \
${WORK_DIR}/aoo40help/${Language}/
```

$\langle \text{UpdateTranslation12 } 23 \rangle$

#### 5.4. ABGLEICH DER EINZELNEN SPRACHEN MIT DEN AKTUELLEN VORLAGEN<sup>23</sup>

Zum Schluss eines jeden Sprachdurchlaufes wird das Git-Repository aktualisiert. Das Ergebnis jeden Durchlaufes resultiert in einem *Commit*.

```
23   $\langle UpdateTranslation12\ 23 \rangle \equiv$  (22)
    # Commands for updating git repository
    cd  ${WORK_DIR}
    git add .
    git commit -am "update $(date) ${Language}"

    echo "Finished "${Language} >>${LOG_FILE}
    echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

    done
fi
```





## Kapitel 6

# Nächtliche Aktivitäten

Das folgende Script, welches nur weitere Skripte aufruft, wird aufgrund eines Cron-Jobs jede Nacht um 2:00 Uh (UTC) ausgeführt.

```
25  <mnrun.sh 25>≡  
    #!/bin/bash  
  
    /home/mechtilde/local/bin/sync-db2files.sh 2>/dev/null  
    #echo "Finished syncing database to files"  
    /home/mechtilde/local/bin/cp2home.sh  
    #/home/mechtilde/local/bin/mksdf.sh
```

## 6.1 Extrahieren aus Pootle

Die Ergebnisse der Übersetzungen werden permanent über die Web-Anwendung in die Pootle-Datenbank eingepflegt.

Das Skript übertägt die Änderungen aus der Datenbank in die *PO*-Dateien.

```
26 <sync-db2files.sh 26>≡
    #!/bin/bash

    ScriptName="sync-db2files" #(.sh)

    # Version 0.1.0

    # This file should be in ~/local/bin/.

    <License 9>

    # This script synchronizes the pootle database to the file system
    # for aoo42 and/or aoo42help.

    <Logging 10>
    <PootleEnv 11>
    <Langs 12>

    for Language in ${LANGS[*]}
    do

        echo ${Language} >>${LOG_FILE}

        # When the --overwrite option is specified, the sync operation will not be
        # conservative and it will overwrite the existing files on disk, making
        # strings obsolete and updating the file's structure.

        echo "UI" >>${LOG_FILE}

        ${VIRTUAL_ENV}/bin/python ${VIRTUAL_ENV}/bin/pootle \
        --config ${VIRTUAL_ENV}/pootle.conf \
        sync_stores --overwrite --project=aoo40 --language=${Language}
```

```
echo "HELP" >>${LOG_FILE}

${VIRTUAL_ENV}/bin/python ${VIRTUAL_ENV}/bin/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
sync_stores --overwrite --project=aoo40help --language=${Language}

echo "Done "${Language} >>${LOG_FILE}

done

echo "Done all"${date} >>${LOG_FILE}
```

## 6.2 Kopieren ins Home-Verzeichnis

Das folgende Skript kopiert die PO-Dateien aus den entsprechenden Unterverzeichnissen des Verzeichnisses */x1/www/po* in die entsprechenden Unterverzeichnisse des Home-Verzeichnisses des Nutzers.

Dort werden gegebenenfalls nicht vorhandene Sprach-Unterverzeichnisse abgelegt. Außerdem fügt das Skript neue Dateien dem Git-Repository hinzu und erzeugt einen Commit.

```
28  <cp2home.sh 28>≡
    #!/bin/bash
    ScriptName="cp2home" #(.sh)

    <License 9>

    # This script copies the *.po files
    # of aoo40 and aoo40help into the home directory

    <Logging 10>

    # Work directory
    WORK_DIR=${HOME}/aoo/workdir

    PROJECT_PATH=/x1/www/po

    <Langs 12>

    for Language in ${LANGS[*]}
    do

        # Language2=$(echo ${Language} | sed -e 's/_/-/1')
        # mkdir -p ${WORK_DIR}/SDF-Files/${Language2}/

        mkdir -p ${WORK_DIR}/PO-Files/${Language}/
```

```
cd ${WORK_DIR}/PO-Files/

cp --recursive ${PROJECT_PATH}/aoo40/${Language} .
cp --recursive ${PROJECT_PATH}/aoo40/${Language} ../aoo40

cp --recursive ${PROJECT_PATH}/aoo40help/${Language}/helpcontent2 \
./${Language}
cp --recursive ${PROJECT_PATH}/aoo40help/${Language} \
../aoo40help/

done

# Commands for updating git repository
cd ${WORK_DIR}
git add .
git commit -am "update $(date) PO-Files"
```



## Kapitel 7

# PO-Dateien nach SDF-Dateien

### Funktioniert erst auf einer aktuellen ASF-Maschine

Auf dem von der ASF bereitgestellten VM gibt es im Home-Verzeichnis eine Kopie der aus der Datenbank extrahierten PO-Dateien in einem Git-Repository.

Dieses wird zur Zeit noch auf einer aktuellen Maschine weiterverarbeitet.

Dazu wird das geclonete Git-Verzeichnis

*mechtilde@translate-ec2-va.apache.org:/home/mechtilde/aoo/workdir* aktualisiert.

Danach wird mithilfe eines Skriptes aus den heruntergeladenen PO-Dateien wieder eine aktualisierte SDF-Datei.

### 7.1 Header-Text

Dazu muss es im Verzeichnis eine Datei *header.txt* geben, die den Lizenztext für die zu generierenden SDF-Dateien enthält.

31 *<header.txt 31>*≡  
#

```

# #####   ###   #   #   ###   #####   #####   #####   #####   #####
#   #   #   #   #   ##   #   #   #   #   #   #   #   #   #   #
#   #   #   #   #   #   #   #   #   #   #   #   #   #   #   #
#   #   #   #   #   #   ##   #   #   #   #   #   #   #   #   #
#   #####   ###   #   #   ###   #   #####   #####   #####   #
#
#   DO NOT EDIT! This file will be overwritten by localization process
#
#####
#
#   Licensed to the Apache Software Foundation (ASF) under one
#   or more contributor license agreements.  See the NOTICE file
#   distributed with this work for additional information
#   regarding copyright ownership.  The ASF licenses this file
#   to you under the Apache License, Version 2.0 (the
#   "License"); you may not use this file except in compliance
#   with the License.  You may obtain a copy of the License at
#
#   http://www.apache.org/licenses/LICENSE-2.0
#
#   Unless required by applicable law or agreed to in writing,
#   software distributed under the License is distributed on an
#   "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
#   KIND, either express or implied.  See the License for the
#   specific language governing permissions and limitations
#   under the License.
#
#####

```



## 7.2 Skript zur Erzeugung der SDF-Dateien

Das folgende Skript erzeugt für jede Sprache aus den *PO*-Dateien eine *SDF*-Datei.

```

33  <po2oo.sh 33>≡
    #!/bin/bash

    ScriptName="po2oo" #(.sh)

    # Version 0.1.0

    # This file should be in ~/local/bin/.

    <License 9>
    <Logging 10>


    # If this script will be able to run on the Pootle server,
    # you have to replace 'DocRoot' by 'WorkDir'


    DocRoot=/home/mechtilde/Projekte/Git/ApacheOpenOffice/A00-PootleWorkdir


    # First, all language files are fetched from
    # translate-ec2-va.apache.org to A00-PootleWorkdir/PO-Files via git on a
    # certain date. There are then all language files.


    <Langs 12>


    for Language in ${LANGS[*]}
    do
        # Replace "_" by "-" for SDF directories
        SDFLANG=$(echo ${Language} | sed -e 's/_/-/')

        echo -e "\n"${Language}" "${SDFLANG}"\n" | tee --append ${LOG_FILE} 2>&1
        echo -e "\n"${Language}" "${SDFLANG}"\n" >>${LOG_FILE}


    <po2oo1 34a>

```

Die *Sdf*-Dateien sollen mit dem Lizenz-Text versehen werden.

34a  $\langle po2001\ 34a \rangle \equiv$  (33)

```
mkdir -p ${DocRoot}/SDF-Files/${SDFLANG}/
cp ${DocRoot}/header.txt ${DocRoot}/SDF-Files/${SDFLANG}/localize.sdf
cd ${DocRoot}/PO-Files/${Language}/
```

$\langle po2002\ 34b \rangle$

Verwendung von *po2oo*:

```
po2oo [options] [--template=en-US.sdf] --languages=<targetlang> \
<input> <output>
```

*targetlang* ist der ISO-639-Code der Zielsprache <sup>1</sup> (s. Kapitel 1.2, Seite 2).

*input* ist das Verzeichnis der übersetzten *PO*-Dateien

*output* ist die daraus erzeugte SDF-Datei.

Als Optionen werden verwendet *--template*, *--skipsource* und *--keeptimestamp*.

Das Template ist eine valide SDF-Datei.

34b  $\langle po2002\ 34b \rangle \equiv$  (34a)

```
po2oo --progress=names \
--template=${DocRoot}/en-US.sdf \
--skipsource \
--keeptimestamp \
--language=${SDFLANG} \
${DocRoot}/PO-Files/${Language}/ \
${DocRoot}/SDF-Files/localize.${SDFLANG} 2>>${LOG_FILE}
```

$\langle po2003\ 35 \rangle$

---

<sup>1</sup>[https://de.wikipedia.org/wiki/ISO\\_639](https://de.wikipedia.org/wiki/ISO_639)

Der Inhalt der erzeugten *SDF*-Datei wird in die zuvor erzeugte Datei *localize.sdf* eingefügt, Nach dieser Zusammenführung wird die von *po2oo* erzeugte (Hilfs-)Datei entfernt.

```
35  <po2oo3 35>≡ (34b)
    cat ${DocRoot}/SDF-Files/localize.${SDFLANG} >>\
    ${DocRoot}/SDF-Files/${SDFLANG}/localize.sdf &&
    rm ${DocRoot}/SDF-Files/localize.${SDFLANG}
    done

    echo "GSICHECK must be done in the build vm."
    echo "The script there is called 'gsicheck.sh'."
```



## Kapitel 8

# Durchführung des *gsicheck*

Voraussetzung für den *gsicheck* ist eine funktionierende Build-Umgebung.

Auch dies ist Teil des Quellcodes

```
gsicheck -c -t -l de <path-to-language>/localize.sdf
```

-c wird die Zeile, die den Fehler enthält, ausgegeben. Es wird also Context zur Fehlermeldung hinzugefügt.

-t bedeutet, dass nur die Übersetzung geprüft wird.

-l gibt die betreffende Sprache an.

37  $\langle$ gsicheck.sh 37 $\rangle \equiv$

```
#!/bin/bash
```

```
ScriptName="po2oo" #(.sh)
```

```
# Version 0.1.0
```

```
# This file should be in ~/local/bin/.
```

```
 $\langle$ License 9 $\rangle$ 
```

```
LOG=${HOME}/log_$(date +%Y%m%d%H%M%S).txt
```

```
touch ${LOG}
```

```
# Mit diesem Script wird nun nach einem Build auf der VM der GSICHECK
```

```
# durchgeführt.
```

```
# Include the Language data
. ${HOME}/configuration/langs.sh
echo ${LANGS} >>${LOG}

cd ${HOME}/openoffice/main/

for Language in ${LANGS[*]}
do
solenv/bin/gsichack -c -t -l ${Language} ../extras/l10n/source/${Language}/localize.sdf
echo ${Language}
read ok
done
```

## Kapitel 9

# Einzelschritte durchführen

### 9.1 sdf nach pot

Wie bereits in Kapitel 5.2 ( Seite 14) beschrieben, muss als nächstes das Arbeitsverzeichnis bereinigt werden.

Dann wird die Datei *en-US.sdf* in die Vorlagedateien (POT-Dateien) umgewandelt werden (Kapitel 5.2, Seite 14).

```
39  <sdf2pot.sh 39>≡
    #!/bin/bash
    ScriptName="sdf2pot" #(.sh)
    # Version 0.1.0
    # This file should be in ~/local/bin/.
    <License 9>
    <Logging 10>
    <PootleEnv 11>
    TEMPLATES_DIR=/x1/www/po/aoo40/templates

    # If the (non-empty) file en-US.sdf is in ${HOME}:
    if [ -s ${HOME}/en-US.sdf ]
    then
    mv ${HOME}/en-US.sdf ${WORK_DIR}/en-US.sdf
```

```
# To be carefull
rm -r ${WORK_DIR}/POT-UI/
rm -r ${WORK_DIR}/POT-HELP/helpcontent2

mkdir -p ${WORK_DIR}/POT-UI
mkdir -p ${WORK_DIR}/POT-HELP

# Convert en-US.sdf to templates
${POOTLE_DIR}/python ${POOTLE_DIR}/oo2po --progress=verbose \
--pot ${WORK_DIR}/en-US.sdf ${WORK_DIR}/POT-UI/

# Move helpcontent2 to POT-HELP to separate HELP from UI
mv ${WORK_DIR}/POT-UI/helpcontent2 ${WORK_DIR}/POT-HELP/
fi
```



## 9.2 Import der POT-Dateien in die Datenbank

s.a. (Kapitel 5.3, Seite 15)

```
41  <sync-pot-files2db.sh 41>≡
    #!/bin/bash

    ScriptName="sync-pot-files2db" #(.sh)

    <License 9>

    <Logging 10>


    # Write *.pot  files into the database
    # This file should be in ~/local/bin/.


    <PootleEnv 11>


    # Copy PotFiles-* into project directory
    cp --recursive --verbose ${WORK_DIR}/POT-UI/* \
    ${PROJECT_PATH}/aoo40/templates/
    cp --recursive --verbose ${WORK_DIR}/POT-HELP/* \
    ${PROJECT_PATH}/aoo40help/templates/


    echo "Copied POT-files into project directory." >>${LOG_FILE}


    # Update the translation templates into the Pootle database


    echo "Update UI-Stores" >>${LOG_FILE}
    ${POOTLE_DIR}/pootle update_stores --project=aoo40 --language=templates


    echo "Update HELP-Stores" >>${LOG_FILE}
    ${POOTLE_DIR}/pootle update_stores --project=aoo40help --language=templates


    echo "Update done" >>${LOG_FILE}
```

```
echo "All done"
```

## 9.3 Sichern der bisherigen Übersetzung

Hierzu wird das Skript *sync-db2files.sh* genommen, das dies auch in der nächtlichen Sicherung durchführt (s. Kapitel 6.1, Seite 26)

## 9.4 Abgleich der Übersetzungen mit der Vorlage

s.a. Kapitel 5.4, Seite 18

```
43  <pomerge.sh 43>≡
    #!/bin/bash
    ScriptName="pomerge" #(.sh)
    <License 9>
    <Logging 10>

    # This script merges the recent po files with the new template
    # and creates new po files.
    # This file should be in ~/local/bin/.

    <PootleEnv 11>
    <Langs 12>

    # Reconcile languages with templates
    for Language in ${LANGS}
    do

        echo "Pomerge UI "${Language} >>${LOG_FILE}
        echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

        ${POOTLE_DIR}/python ${POOTLE_DIR}/pomerge \
        --template=${PROJECT_PATH}/aoo40/templates/ \
        --input ${WORK_DIR}/aoo40/${Language}/ \
        --output ${PROJECT_PATH}/aoo40/${Language}/
```

```
echo "Pomerge HELP "${Language} >>${LOG_FILE}
echo $(date +%Y-%m-%d-%H%M%S) >>${LOG_FILE}

${POOTLE_DIR}/python ${POOTLE_DIR}/pomerge \
--template=${PROJECT_PATH}/aoo40help/templates/ \
--input ${WORK_DIR}/aoo40help/${Language}/ \
--output ${PROJECT_PATH}/aoo40help/${Language}/

done
```

## 9.5 Datenbank aktualisieren

Anhand der neuen Po-Dateien wird die Datenbank unter `translate.apache.org` aktualisiert, sodass die Übersetzer die Übersetzungen aktualisieren können.

45 `<sync-po-files2db.sh 45>≡`

```
#!/bin/bash

ScriptName="sync-po-files2db" #(.sh)

<License 9>

<Logging 10>

# Write *.po files into the database
# This file should be in ~/local/bin/.

<PootleEnv 11>

<Langs 12>

# Reconcile languages with templates
for Language in ${LANGS[*]}
do
echo ${Language} >>${LOG_FILE}
${VIRTUAL_ENV}/bin/python ${VIRTUAL_ENV}/bin/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
update_stores --project=aoo40 --language=${Language} &&

echo ${Language}"-Help" >>${LOG_FILE}

${VIRTUAL_ENV}/bin/python ${VIRTUAL_ENV}/bin/pootle \
--config ${VIRTUAL_ENV}/pootle.conf \
update_stores --project=aoo40help --language=${Language}
done
```

Als Ergebnis der Ausführung dieses Skriptes sind die PO-Datei und die Datenbank auf dem gleichen (aktuellen) Stand.

## 9.6 Aktualisierung des Arbeitsverzeichnisses

Schließlich wird auch das Arbeitsverzeichnis im Home-Verzeichnis des Nutzers aktualisiert, indem die PO-Dateien mit dem Skript *cp2home.sh* (Kapitel 6.2, Seite 28) dorthin kopiert und mit Git versioniert werden.

# Kapitel 10

## Selten benötigte Skripte

Hier werden die Skripte dokumentiert, die nur beim Aufsetzen eines neuen Server benötigt werden.

- Erstellen von Sprachverzeichnissen
- Wiederherstellung nach fehlerhaftem Abgleich

### 10.1 Erstellen von Sprachverzeichnissen

Mit dem folgenden Skript werden für Sprachen Verzeichnisse im Bereich des Pootle-Servers erstellt.

```
47 <mklangdirs.sh 47>≡  
    #!/bin/bash  
    <License 9>  
  
    # You can use this script to create all language  
    # directories in /x1/www/po/ on a new server.  
    # (if not exist)  
  
    <PootleEnv 11>  
    <Langs 12>
```

```

cd ${PROJECT_PATH}/aoo40

mkdir --verbose ${LANGS}

ls -la

echo "Done in /x1/www/po/aoo40"

cd ${PROJECT_PATH}/aoo42help

mkdir --verbose $(find ${WORK_DIR}/PO-Files/ -name '*' \
| sed --expression='s/\home/mechtilde/aoo/workdir/PO-Files\\\\' \
| grep helpcontent2 | sed --expression='s/\[a-z 0-9\]*.po$//')

```

*⟨mklangsdirs1 48⟩*

Es gibt für die Text der Hilfe unterhalb des Sprachverzeichnisses ein Verzeichnis *helpcontent2*. In *x1/www/po* befinden sich die Hilfetext im Verzeichnis *aoo40help*.

48    *⟨mklangsdirs1 48⟩* ≡ (47)

```

ls -la

echo "Done in /x1/www/po/aoo40help"

echo -e 'Please become root and adapt the rights\nusing "chown -R www-data ."'

```



## 10.2 Wiederherstellung nach Fehler

```

49  <reset-pootle.sh 49>≡
    #!/bin/bash

    ScriptName="reset-pootle" #(.sh)

    # Version 0.1.0

    # This file should be in ~/local/bin/

    <License 9>


    # You can use this script to restore pootle
    # e.g after a wrong import of templates
    # It does it from home directory into the project path


    <Logging 10>


    <PootleEnv 11>
    <Langs 12>


    for Language in ${LANGS}
    do
        echo ${Language} >>${LOG_FILE}
        cp --recursive --verbose ${WORK_DIR}/PO-Files/${Language} \
        ${PROJECT_PATH}/aoo40

        rm --recursive --verbose \
        ${PROJECT_PATH}/aoo40help/${Language}/helpcontent2

        mv --verbose ${PROJECT_PATH}/aoo40/${Language}/helpcontent2 \
        ${PROJECT_PATH}/aoo40help/${Language}

    done


    echo "You should run sync-po-files2db.sh now!"

```

Danach wird das Skript *sync-po-files2db.sh* aufgerufen.

### 10.3 Wiederherstellung der PO-Dateien

Das folgende Skript *sync-po-files.sh* nutzt die PO-Dateien im Arbeitsverzeichnis, um die PO-Dateien auf dem Pootle-Server wiederherzustellen. Es kann auch benutzt werden, um dort die PO-Dateien für alle Sprachen anzulegen. Es werden auf dem Pootle-Server nur die Dateien erneuert die ein älteres Datum haben als im Arbeitsverzeichnis.

```
50  <sync-po-files.sh 50>≡
    #!/bin/bash
    ScriptName="sync-po-files" #(.sh)

    <License 9>

    # You can use this script to create all language
    # directories in /x1/www/po/ on a new pootle server
    # (if not exist) and sync *.po files.

    <PootleEnv 11>

    # Go to the directory where the *.po files are in subdirectories
    cd ${WORK_DIR}/PO-Files

    # Create list of all *.po files in subdirectories helpcontent2
    find . -name '*' | grep helpcontent2 >helpcontent2.txt

    # Copy all files from list to aoo42help
    rsync --verbose --files-from=helpcontent2.txt . ${PROJECT_PATH}/aoo42help/

    echo "Done in "${PROJECT_PATH}"/aoo42help"
```

```
rm --verbose helpcontent2.txt

# Copy all other *.po files
rsync --archive --verbose --exclude='*helpcontent2*' . ${PROJECT_PATH}/aoo42

echo "Done in "${PROJECT_PATH}"/aoo42"

echo -e 'Please become root and adapt the rights\nusing "chown -R www-data .'
```



# Kapitel 11

## Extraktionsskripte

### 11.1 Extraktion der Dokumentation

```
53  <create-pdf.sh 53>≡
    #!/bin/bash

    # With this script you can create a *.pdf and an *.epub file
    # from a *.nw file.
    # In line 6 you can find the file name.

    FileName=TranslationA00

    noweave -index -delay ${FileName}.nw >${FileName}.tex
    noweave -filter l2h -index -html ${FileName}.nw | htmltoc >${FileName}.html
    [ ! -f ${FileName}.idx ] || makeindex ${FileName}.idx

    for ((i=4; i>0; i--))
    do
        echo $i
        pdflatex ${FileName}.tex
    done

    tex4ebook -f epub ${FileName}.tex

    <create-scripts 54>
```

## 11.2 Extraktion der dokumentierten Skripte

54  $\langle create-scripts\ 54 \rangle \equiv$  (53)

```
# Now it extracts the scripts

set -e

scripts="
License
Logging
PootleEnv
LanguageLists.sh
UpdateTranslation.sh
mnrun.sh
sync-db2files.sh
cp2home
header.txt
po2oo.sh
Download.sh
sdf2pot.sh
sync-pot-files2db.sh
reset-pootle.sh
sync-po-files.sh
"

FileName=TranslationA00

t='date +%c'

function build_script()
{
notangle -R"$script" ${FileName}.nw >"$script"
```

```
sed --in-place \  
--expression='s/This is the end, my friend[[:cntrl:]]*/This is the end, my friend/' \  
"$1"  
echo "#generated on $t" >>"$1"  
chmod ugo+x "$1"  
bash -n "$1"  
}  
  
for script in $scripts; do  
  build_script "$script"  
done
```





## Kapitel 12

# Skripte für die Automatisierung

### 12.1 Pootle.conf

```
57  <Pootle.conf 57>≡
    # -*-coding: utf-8 -*-

    """Sample configuration file.

    This file includes the settings that administrators will likely change.
    You can find the defaults in the ‘*.conf’ files for more advanced settings.
    """

    from pootle.i18n.gettext import ugettext_lazy as _

    #
    # Base
    #

    # Site title
```

```
POOTLE_TITLE = _(u'Pootle Translation Server')

# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# although not all choices may be available on all operating systems.
# If running in a Windows environment this must be set to the same as your
# system time zone.
TIME_ZONE = 'UTC'

# Make this unique, and don't share it with anybody.
# SECRET_KEY = '5B9vwlJXdCp/DR0i8i8F2VmyKEKqeTKlnXovP9NLYfbJDdqrnTWhFmRUaCTahthEjNA='
SECRET_KEY = 'FpuS2TZuWjID9XShjBN8ngivUPqo30QemCBJgH1lDd2fjyj9GtEYCdmisN5guDZFEgs='

# A list of strings representing the host/domain names that this Pootle server
# can serve. This is a Django's security measure.
ALLOWED_HOSTS = [
    '127.0.0.1',
    'localhost',
    'translate.apache.org',
    'translate-ec2-va.apache.org',
    #'${your_server}',
]

# Set this to a canonical url for your site *without trailing slash*
# This will be used when deriving urls to send out emails
# If you use the 'django.contrib.sites' framework set this to blank
POOTLE_CANONICAL_URL = "https://translate.apache.org"

#
# Backends
#
```

```

# Database backend settings
DATABASES = {
    'default': {
        # Replace 'sqlite3' with 'postgresql' or 'mysql'.
        # 'ENGINE': 'django.db.backends.sqlite3',
        'ENGINE': 'django.db.backends.mysql',
        # Database name or path to database file if using sqlite3.
        # 'NAME': working_path('dbs/pootle.db'),
        'NAME': 'pootledb',
        # Not used with sqlite3.
        # 'USER': "",
        'USER': 'pootle',
        # Not used with sqlite3.
        # 'PASSWORD': "",
        'PASSWORD': 'x2vv5YbYjy0L16vrR504pdjJz+bJd5R0',
        # Set to empty string for localhost. Not used with sqlite3.
        # 'HOST': "",
        'HOST': '127.0.0.1',
        # Set to empty string for default. Not used with sqlite3.
        'PORT': "",
        # See https://docs.djangoproject.com/en/1.10/topics/db/transactions/
        # required for Django + sqlite
        'ATOMIC_REQUESTS': True,
        'OPTIONS' : {
            # # MySQL:
            # # https://docs.djangoproject.com/en/1.10/ref/databases/#setting-sql-mode
            # 'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
        },
        'TEST': {
            'NAME': 'test_pootle',
        }
    }
}

```

```
}

# Cache Backend settings

# For more information, check
# https://docs.djangoproject.com/en/1.10/topics/cache/#setting-up-the-cache
# and http://niwibe.github.io/django-redis/
#CACHES = {
#    'default': {
#        'BACKEND': 'django_redis.cache.RedisCache',
#        'LOCATION': 'redis://127.0.0.1:6379/1',
#        'TIMEOUT': 60,
#    },
#    'redis': {
#        'BACKEND': 'django_redis.cache.RedisCache',
#        'LOCATION': 'redis://127.0.0.1:6379/2',
#        'TIMEOUT': None,
#    },
#    'lru': {
#        'BACKEND': 'django_redis.cache.RedisCache',
#        'LOCATION': 'redis://127.0.0.1:6379/3',
#        'TIMEOUT': 604800, # 1 week
#    },
#    'exports': {
#        'BACKEND': 'django.core.cache.backends.filebased.FileBasedCache',
#        'LOCATION': working_path('exports/'),
#        'TIMEOUT': 259200, # 3 days.
#    },
#}

#
```

```
# Local Translation Memory service
#

# Setup for connection to the ElasticSearch server for translation memory based
# on the projects hosted on Pootle.
# You may want to set AMAGAMA_URL to " (empty string) if using this service.
#
#POOTLE_TM_SERVER = {
#    'local': {
#        'ENGINE': 'pootle.core.search.backends.ElasticSearchBackend',
#        'HOST': 'localhost',
#        'PORT': 9200,
#        # Every TM server must have its own unique index.
#        'INDEX_NAME': 'translations',
#        # Provides a weighting factor to alter the final score for TM results
#        # from this TM server. Valid values are between '0.0' and '1.0',
#        # both included. Defaults to '1.0' if not provided.
#        'WEIGHT': 1.0,
#    },
#    'external': {
#        'ENGINE': 'pootle.core.search.backends.ElasticSearchBackend',
#        'HOST': 'localhost',
#        'PORT': 9200,
#        'INDEX_NAME': 'external-translations',
#        'WEIGHT': 0.9,
#    },
#}

#
# Logging
#
```

```
# The directory where Pootle writes its logs
POOTLE_LOG_DIRECTORY = '/x1/www/pootle/log'

#
# Site
#

# This Pootle server admins
ADMINS = (
    ('Chris Lambertus', 'cml@apache.org'),
    # ('Your Name', 'your_email@domain.com'),
)

# Mail settings

# Default email address to use for messages sent by Pootle.
DEFAULT_FROM_EMAIL = 'translate@apache.org'

# Address to receive messages sent by contact form.
POOTLE_CONTACT_ENABLED = False
POOTLE_CONTACT_EMAIL='root@apache.org'

# Email address to report string errors to, unless a report email was set for
# the project for which the string error is being reported.
# Use this as a way to forward string error reports to an address
# different from the general contact address.
POOTLE_CONTACT_REPORT_EMAIL = 'users@infra.apache.org'

# Whether to email reviewer's feedback to suggesters.
POOTLE_EMAIL_FEEDBACK_ENABLED = False
```

```
# Mail server settings

# By default Pootle uses the SMTP server on localhost. If the server is not
# configured for sending emails, uncomment and use these settings to setup an
# external outgoing SMTP server.

# Example for Google as an external SMTP server
#EMAIL_HOST_USER = 'USER@YOUR_DOMAIN.com'
#EMAIL_HOST_PASSWORD = 'YOUR_PASSWORD'
#EMAIL_HOST = 'smtp.gmail.com'
#EMAIL_PORT = 587
#EMAIL_USE_TLS = True

#

# Translation

#

# The directory where the translation files are kept
# POOTLE_TRANSLATION_DIRECTORY = working_path('translations')
POOTLE_TRANSLATION_DIRECTORY = '/x1/www/po'

# Two-tuple defining the markup filter to apply in certain textareas.
#
# -Accepted values for the first element are 'markdown' and
#   'html' (deprecated).
# -The second element should be a dictionary of keyword arguments that will be
#   passed to the markup function.
#
POOTLE_MARKUP_FILTER = ('markdown', {})

# Set the backends you want to use to enable translation suggestions through
```

```

# several online services. To disable this feature completely just comment all
# the lines to set an empty list [] to the POOTLE_MT_BACKENDS setting.
#
# The second parameter for each backend option is the API key, which will
# be used in case the service supports using an API key.
#
# Available options are:
# 'CAIGHDEAN_TRANSLATE': Caighdeán Translate service.
#
#         Provides translation from Manx Gaelic (gv) and
#         Scottish Gaelic (gd), into Irish Gaelic (ga-IE)
#
#         See more at https://github.com/kscanne/caighdean
# 'GOOGLE_TRANSLATE': Google Translate service.
#
#         For this service you need to set the API key.
#
#         Note that Google Translate API is a paid service
#
#         See more at https://cloud.google.com/translate/v2/pricing
# 'WELSH_TRANSLATE': Welsh Translate service.
#
#         Provides translation from English (en) to Welsh (cy)
#
#         For this service you need an API key.
#
#         See more at http://techiaith.cymru/api/translation/?lang=en
# 'YANDEX_TRANSLATE': Yandex Translate service.
#
#         For this service you need to set the API key.
#
#         See more at https://tech.yandex.com/translate/
#
POOTLE_MT_BACKENDS = [
#     ('CAIGHDEAN_TRANSLATE', ""),
#     ('GOOGLE_TRANSLATE', ""),
#     ('WELSH_TRANSLATE', ""),
#     ('YANDEX_TRANSLATE', ""),
]

# Custom templates directories

```



```
#
TEMPLATES[0]['DIRS'] = [
# working_path(os.path.join('custom', 'templates')),
]

POOTLE_SIGNUP_ENABLED = False
AUTH_LDAP_USER_QUERY_FIELD = 'username'
AUTH_LDAP_ALWAYS_UPDATE_USER = True
AUTH_LDAP_SERVER_URI = 'ldaps://ldap-master.apache.org'
AUTH_LDAP_BASE_DN = 'ou=people,dc=apache,dc=org'
AUTH_LDAP_BIND_DN = 'cn=nss_ldap,ou=users,ou=services,dc=apache,dc=org'
AUTH_LDAP_BIND_PASSWORD = 'b1t3me'
AUTH_LDAP_USER_DN_TEMPLATE = 'uid=%(user)s,ou=people,dc=apache,dc=org'
AUTHENTICATION_BACKENDS = (
'django_auth_ldap.backend.LDAPBackend',
'django.contrib.auth.backends.ModelBackend',
)
AUTH_LDAP_USER_ATTRLIST = ["*", "+"]
AUTH_LDAP_USER_ATTR_MAP = {
'first_name': 'givenName',
'last_name': 'sn',
'email': 'asf-committer-email',
'username': 'uid',
'full_name': 'cn',
}
```

<https://ci2.apache.org/#/builders/70/builds/43>

<https://svn.apache.org/repos/infra/infrastructure/buildbot2/projects/openofficeorg.py>

<https://ci2.apache.org/#/builders/70>

<https://nightlies.apache.org/openoffice/install/linsnap-42x/?C=M;O=D>

<https://github.com/translate/pootle/issues/6902>

<https://github.com/mozilla/pontoon>

<https://github.com/WeblateOrg/weblate/discussions>