How to build backend with scripts

How to build backend with scripts

Cette note propose quelques scripts de gestion pour réaliser un build du backend d'*Aero-Web* et l'installer dans *Tomcat*.

Prerequisites

- Mise en oeuvre du Backend Aero-Webb

Scripts Linux

Les scripts présentés ici ne sont exécutables que dans l'environnement Linux, sous WSL.

Versions non définitives

Les scripts de cette page sont largement améliorables ; ils ne sont ici au titre de la capitalisation

01-do_git_update_aw_backend.sh

Ce script récupère la dernière version des sources des projets *Service*, *GUI Std controllers* et *Webapp*.

```
#!/bin/bash

AW_WORKSPACE_ROOT_DIR=${HOME}/Workspaces/Aero-Webb
AW_BACKEND_ROOT_DIR=${AW_WORKSPACE_ROOT_DIR}/Backend
BASE_DIR=$(pwd)

echo "##### Update Aero-Webb backend projects from Git repositories..."
echo
echo "##-> Fetch/Pull Backend:services..."
echo
cd ${AW_BACKEND_ROOT_DIR}/aero-webb-services
git fetch --prune
git pull
echo

echo "##-> Fetch/Pull Backend:gui-std-controllers..."
echo
cd ${AW_BACKEND_ROOT_DIR}/gui-std-controllers
git fetch --prune
git pull
echo

echo "##-> Fetch/Pull Backend:webapp..."
echo
cd ${AW_BACKEND_ROOT_DIR}/webapp
git fetch --prune
git pull
echo

echo "##### Job done."
cd ${BASE_DIR}
```

Puis, rendre le script exécutable :

```
chmod +x 01-do_git_update_aw_backend.sh
```

02-do_build_aw_backend.sh

Ce script construit les projets *Aero-Webb* : *Service*, *GUI Std controllers* et *Webapp*.

```
#!/bin/bash

AW_WORKSPACE_ROOT_DIR=${HOME}/Workspaces/Aero-Webb
AW_BACKEND_ROOT_DIR=${AW_WORKSPACE_ROOT_DIR}/Backend
BASE_DIR=$(pwd)

echo "##### Build Aero-Webb backend projects..."
echo
echo "##-> Build Backend:services..."
echo
cd ${AW_BACKEND_ROOT_DIR}/aero-webb-services
mvn clean install -P postgres,obfuscation -DskipTests
echo

echo "##-> Build Backend:gui-std-controllers..."
echo
cd ${AW_BACKEND_ROOT_DIR}/gui-std-controllers
mvn clean install -U -DskipTests
echo

echo "##-> Build Backend:webapp..."
cd ${AW_BACKEND_ROOT_DIR}/webapp
mvn clean package -U
echo

echo "##### Job done."
cd ${BASE_DIR}
```

Puis, rendre le script exécutable :

```
chmod +x 02-do_build_aw_backend.sh
```

03-deploy_war_in_tomcat.sh

Ce script déploie le WAR du projet *Webapp* dans le serveur *Tomcat*.

> ⚠ **Adaptation à faire**
>
> Avant d'exécuter le script, ne pas oublier de modifier la valeur de la variable USER_HOME avec son propre compte Linux.

```
#!/bin/bash
#
# WARNING : This script must be executed as root user (or with 'sudo' command).
#
USER_HOME=/home/lambda
AW_WORKSPACE_ROOT_DIR=${USER_HOME}/Workspaces/Aero-Webb
AW_BACKEND_ROOT_DIR=${AW_WORKSPACE_ROOT_DIR}/Backend
TOMCAT_ROOT_DIR=/opt/tomcat/latest
TOMCAT_PID_FILE=${TOMCAT_ROOT_DIR}/tomcat.pid

if [ "$EUID" -ne 0 ]; then
 echo "Please run this script as root (or with 'sudo' command)."
 exit 1
fi

if [ -f $TOMCAT_PID_FILE ]; then
 TOMCAT_PID=$(cat $TOMCAT_PID_FILE)
 if ![ ps -p $TOMCAT_PID ]; then
 rm -f $TOMCAT_PID_FILE
 fi
fi

BASE_DIR=$(pwd)

echo "##### Aero-Web WAR installation..."
echo
service tomcat stop
echo

echo "##-> Cleaning old WAR file..."
if [ -f ${TOMCAT_ROOT_DIR}/webapps/aerowebb.war ]; then
 rm -f ${TOMCAT_ROOT_DIR}/webapps/aerowebb.war
fi

if [ -d ${TOMCAT_ROOT_DIR}/webapps/aerowebb ]; then
 rm -rf ${TOMCAT_ROOT_DIR}/webapps/aerowebb
fi

echo "##-> Deploying WAR file..."
cp ${AW_BACKEND_ROOT_DIR}/webapp/target/aerowebb.war ${TOMCAT_ROOT_DIR}/webapps/
chown tomcat:tomcat ${TOMCAT_ROOT_DIR}/webapps/aerowebb.war

echo
service tomcat start
echo
echo "##### Job done."
```

Puis, rendre le script exécutable :

```
chmod +x 03-deploy_war_in_tomcat.sh
```

04-download_db_daily_backup.sh

Ce script permet de télécharger la dernière sauvegarde (dump) de la base de données du daily standard.

Prerequisites

Avoir installé l'outil "curl", via la commande suivante :

```
sudo apt install curl
```

Mot de passe

Le mot de passe stocké dans la variable FTP_PASS n'est qu'un exemple.

Demander au service IT la valeur à renseigner.

```
#!/bin/bash

FILENAME_PREFIX=daily_std_build
FILENAME_SUFFIX=backup
INPUT_ROOT_DIR=NEW
INPUT_DIR=${INPUT_ROOT_DIR}/BDD-VM-2M-DAILY-STD-BUILD
OUTPUT_DIR=${HOME}/db_backup
FTP_HOST="ftp://ftp2.2moro.fr/${INPUT_DIR}/"
FTP_USER="BDD"
FTP_PASS='abcdefghijkl'

echo "##### Last database daily backup download"
echo

LAST_BACKUP_FILENAME=$(curl -s -u "${FTP_USER}:${FTP_PASS}" --ssl-reqd  ${FTP_HOST}/ | cut -b 59-
| grep -E "${FILENAME_PREFIX}_.*\.${FILENAME_SUFFIX}" | sort -r | head -1)

if [ -z "${LAST_BACKUP_FILENAME}" ]; then
  echo "Error: no database dump file found to download."
  exit 1
fi

echo "Downloading '${LAST_BACKUP_FILENAME}' into '${OUTPUT_DIR}' directory..."
echo

mkdir -p ${OUTPUT_DIR}
curl -u "${FTP_USER}:${FTP_PASS}" --ssl-reqd  ${FTP_HOST}/${LAST_BACKUP_FILENAME} -o
${OUTPUT_DIR}/${LAST_BACKUP_FILENAME}

echo
echo "##### Job done."
```

Puis, rendre le script exécutable :

```
chmod +x 04-download_db_daily_backup.sh
```

05-update_db_with_daily_backup.sh

Ce script permet de mettre à jour la base de données de son environnement local avec la suavegarde téléchargée à l'aide du script précédent (cf. 04-download_db_daily_backup.sh).

```
#!/bin/bash

POSTGRESQL_DB="aerowebb"
POSTGRESQL_USER="postgres"
POSTGRESQL_PASS="postgres"
PGPASSWORD=${POSTGRESQL_PASS}

TODAY=$(date '+%Y-%m-%d')
INPUT_DIR=$HOME/db_backup
INPUT_FILE=${INPUT_DIR}/daily_std_build_${TODAY}.backup

# Close opened database connections.
TERMINATE_CONNECTIONS="\
SELECT pg_terminate_backend(pg_stat_activity.pid) \
FROM pg_stat_activity \
WHERE pg_stat_activity.datname = '${POSTGRESQL_DB}' \
 AND pid <> pg_backend_pid(); \
"

if [ ! -r ${INPUT_FILE} ]; then
 echo "ERROR: File '${INPUT_FILE}' does not exist or not readable."
 exit 1
fi

echo "##### Database updating..."
echo

echo "##-> Terminate DB connections..."
echo ${TERMINATE_CONNECTIONS}
psql -U ${POSTGRESQL_USER} -c "$TERMINATE_CONNECTIONS"

echo "##-> DROP database..."
echo "psql -U ${POSTGRESQL_USER} -c \"DROP DATABASE IF EXISTS ${POSTGRESQL_DB};\""
psql -U ${POSTGRESQL_USER} -c "DROP DATABASE IF EXISTS ${POSTGRESQL_DB};"

echo "##-> CREATE database..."
echo "psql -U ${POSTGRESQL_USER} -c \"{CREATE DATABASE $POSTGRESQL_DB};\""
psql -U ${POSTGRESQL_USER} -c "CREATE DATABASE ${POSTGRESQL_DB};"

echo "##-> Restore DB from backup..."
echo "pg_restore -U ${POSTGRESQL_USER} -d ${POSTGRESQL_DB} ${INPUT_FILE}"
pg_restore -U ${POSTGRESQL_USER} -d ${POSTGRESQL_DB} ${INTPUT_FILE}

echo
echo "##### Job done."
```

Puis, rendre le script exécutable :

```
chmod +x 05-update_db_with_daily_backup.sh
```