Migrate DSpace-Metadata from DSpace to VIVO

Draft

Page in draft form

Summary

Introduction

This page presents the procedure for migrating data from DSpace to VIVO. It answers the use case of a VIVO instance in read-only mode used to present the metadata contained in DSpace

Goals

- The scenario to be realized by this procedure consists in developing the necessary steps in order to migrate the metadata of two DSpace instances (the DSpace-6 Demo instance and the DSpace-7 Demo instance) to a local VIVO instance
- At the end of this procedure, the experimenter should have a fully operational VIVO instance containing the metadata harvested from DSpace-6-Demo DSpace-7-Demo, both of which are available from the web.
- The experimenter will also have in his possession, the necessary information to harvest in VIVO the metadata of a DSpace instance that he will have chosen and that it is possible to harvest from an OAI-PMH endpoint

Useful addresses

Title	URL	Description
DSpace-6 Demo Home Page	https://demo.dspace.org/	This entry page links to other links concerning the DSpace-6 demo
DSpace-6 Demo UI	https://demo.dspace.org/xmlui/	This page is the DSpace-6 api allowing to manipulate metadata
DSpace-6 OAI Api	https://demo.dspace.org/oai/request	OAI API used to harvest data
DSpace-7 Demo Home Page	https://demo7.dspace.org/home	This entry page links to other links concerning the DSpace-7 demo
VIVO Project GitHub Home Page	https://github.com/vivo-project	Source code location needed to install VIVO
DSpace-VIVO Integration Project (DV-IP)	https://github.com/vivo-community/DSpace-VIVO	Source code location for the migration of DSpace metadata to VIVO
DSpace-VIVO ETL Example	https://github.com/vivo-community/DSpace-VIVO/tree/main/test/org.vivoweb.dspacevivo.etlexample	Source code location for extract-transform-load (ETL) metadata processing from DSpace to VIVO

Useful variable and constant names

Title	Var Name	Var Value	Description
Project root directory	DVIP_HOME_PRJ	~/dspace-vivo-prj	The value content is a suggestion
Git root directory	GIT_REPO	$DVIP_HOME_PRJ/00-GIT	Directory containing extracted GIT projects
Default VIVO login (username - password)	admin@vivo.org	Vivo1234.	To be used to log-in as a VIVO administrator
local server URLs	SOLR	http://localhost:8983/solr/#/	
	VIVO	http://localhost:8080/vivo-dspace/	

Software requirements

- jdk 11
- maven 3.6.3
- Linux Ubuntu
- No **solr** or **tomcat** instance should be running on the computer
- Linux bash

Setting up

Setting up the necessary resources for running VIVO

Step name and description	Commands
Setting up project	`mkdir -p ~/dspace-vivo-prj/00-GIT` `cd ~/dspace-vivo-prj/00-GIT`
Retrieve the DV-IP source code	`git clone --depth 1 --branch Beta-1.1 https://github.com/vivo-community/DSpace-VIVO`
Install Solr + Tomcat	`./DSpace-VIVO/releng/org.vivoweb.dspacevivo.installer/00-INIT/install-tomcat-solr-app.sh`
Installing/compiling VIVO	`./DSpace-VIVO/releng/org.vivoweb.dspacevivo.installer/01-VIVO/vivo-git-clone.sh` `./DSpace-VIVO/bundles/org.vivoweb.dspacevivo/script/vivo-compile-and-deploy-for-tomcat.sh`

Run - Start/Stop VIVO	**Starting VIVO**
	`cd ./DSpace-VIVO/bundles/org.vivoweb.dspacevivo/script`
	`source ./00-env.sh`
	`solr-start.sh`
	`tomcat-start.sh`
	
	To show VIVO in a Web Browser (http://localhost:8080/vivo-dspace/)
	`browse-vivo.sh`
	
	For stopping VIVO
	`tomcat-stop.sh`
	`solr-stop.sh`

Setting up the necessary resources for running Dspace

Installing the migration utilities

Step name and description	Commands
Install Apache Jena and its other associated tools	`./DSpace-VIVO/releng/org.vivoweb.dspacevivo.installer/99-OTHER_TOOLS/jena-git-clone-and-deploy.sh`
Compiling/Installing DSpace-VIVO-EXEMPLE and its code libraries	`./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/script/mvn_install_example.sh`

Confirm the installation

The purpose of this step is to validate the correct installation of the components necessary for the scenario to proceed. Here is a series of command that can be executed along with their execution result allowing you to compare them with the result of your own installation

Step name and description	Commands

Validate that the OS contains all the necessary commands to run the dspace2vivo scripts	**Run the script to validate the required applications being installed**
	`./DSpace-VIVO/releng/org.vivoweb.dspacevivo.installer/99-OTHER_TOOLS/validate-syscmd-config.sh`
	
	Result summary
	`adduser ok!`
	`ant ok!`
	`as ok!`
	`at ok!`
	`awk ok!`
	`basename ok!`
	`bash ok!`
	`cat ok!`
	`chmod ok!`
	`chown ok!`
	`chroot ok!`
	`clear ok!`
	`convert ok!`
	`cp ok!`
	`curl ok!`
	`cut ok!`
	`...`
	
	To present the applications to be installed
	`./DSpace-VIVO/releng/org.vivoweb.dspacevivo.installer/99-OTHER_TOOLS/validate-syscmd-config.sh  | grep NOT`

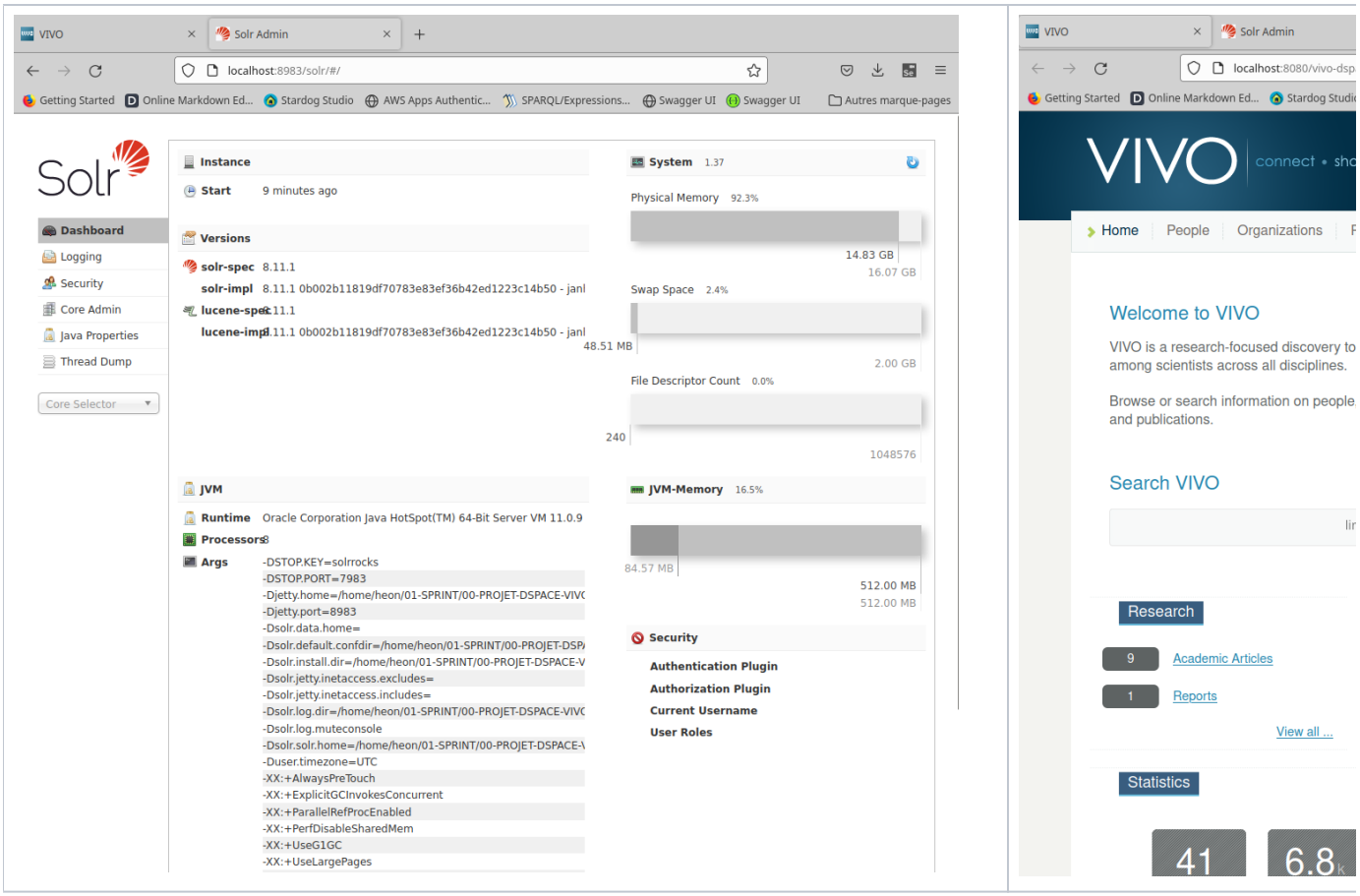
To identify the package to install
for a given application, simply type the command on the command line and run the proposal offered by the system
	Example
	`$ as`
	
	`Command 'as' not found, but can be installed with:`
	
	`sudo apt install binutils`
	
	`ubuntu@ip-172-22-10-100:~/dspace-vivo-prj/00-GIT$ sudo apt install binutils`
	`Reading package lists... Done`
	`Building dependency tree`
	`Reading state information... Done`
	`The following additional packages will be installed:`

Validate that all necessary GIT projects are cloned and properly deployed	**Excute 'ls' command from $GIT_REPO**
	`ls -l`
	`total 24`
	`drwxrwxr-x  6 heon heon 4096 mai 20 14:04 data-format-translator`
	`drwxrwxr-x  7 heon heon 4096 mai 20 11:02 DSpace-VIVO`
	`drwxrwxr-x  9 heon heon 4096 mai 20 11:08 Vitro`
	`drwxrwxr-x 11 heon heon 4096 mai 20 11:08 Vitro-languages`
	`drwxrwxr-x 10 heon heon 4096 mai 20 11:08 VIVO`
	`drwxrwxr-x 11 heon heon 4096 mai 20 11:08 VIVO-languages`
	
	Execute 'ls' from $GIT_REPO in deploy directory
	`ls -dl ./DSpace-VIVO/deploy/*/`
	`drwxrwxr-x 9 heon heon 4096 mai 20 11:07 ./DSpace-VIVO/deploy/app-solr/`
	`drwxrwxr-x 9 heon heon 4096 mai 20 11:07 ./DSpace-VIVO/deploy/app-tomcat/`
	`drwxrwxr-x 2 heon heon 4096 mai 20 14:05 ./DSpace-VIVO/deploy/lib/`
	`drwxrwxr-x 7 heon heon 4096 mai 20 14:04 ./DSpace-VIVO/deploy/translator/`
	`drwxrwxr-x 9 heon heon 4096 mai 20 11:13 ./DSpace-VIVO/deploy/vivo-home/`

Test the utilities to make sure they are working	**Setting up environment variables in your session (From $GIT_REPO)**
	`source ./DSpace-VIVO/bundles/org.vivoweb.dspacevivo/script/00-env.sh`
	
	Validate Solr
	`solr-start.sh`
	
	`Waiting up to 180 seconds to see Solr running on port 8983 [|]`
	`Started Solr server on port 8983 (pid=1741315). Happy searching!`
	
	`solr-status.sh`
	
	`Found 1 Solr nodes:`
	`Solr process 56366 running on port 8983`
	`{`
	`  "solr_home":"xxxxxxx/00-GIT/DSpace-VIVO/deploy/app-solr/server/solr",`
	`  "version":"8.11.1 0b002b11819df70783e83ef36b42ed1223c14b50 - janhoy - 2021-12-14 13:50:55",`
	`  "startTime":"2022-05-19T15:15:10.534Z",`
	`  "uptime":"0 days, 17 hours, 25 minutes, 10 seconds",`
	`  "memory":"151 MB (%29.5) of 512 MB"}`
	
	Validate Tomcat
	`tomcat-start.sh`
	
	`Using CATALINA_BASE:  xxxxxxx/00-GIT/DSpace-VIVO/deploy/app-tomcat`
	`Using CATALINA_HOME:  xxxxxxx/00-GIT/DSpace-VIVO/deploy/app-tomcat`
	`Using CATALINA_TMPDIR: xxxxxxx/00-GIT/DSpace-VIVO/deploy/app-tomcat/temp`
	`Using JRE_HOME:  /opt/jdk-11.0.9`
	`Using CLASSPATH:  xxxxxxx/00-GIT/DSpace-VIVO/deploy/app-tomcat/bin/tomcat-juli.jar`
	`Using CATALINA_OPTS:`
	`Tomcat started.`
	
	Test Apache-Jena
	`sparql -version 2>/dev/null`
	
	`Jena:  VERSION: 3.17.0`
	`Jena:  BUILD_DATE: 2020-11-25T19:40:23+0000`

Visual confirmation in your web browser

Visual of Solr	Visual for VIVO-DSpace
URL = http://localhost:8983/solr/#/	**URL = http://localhost:8080/vivo-dsp**



Migrate data from DSpace6-Demo and DSpace-7-Demo into VIVO with default migration values

- This scenario performs the DSpace Items reading of the Demo-DSpace 6 (https://demo.dspace.org/jspui/) and DemoDSpace-7 (https://demo7.dspace.org/) demonstration sites.
- In order to achieve a complete extraction in a respectable time, the data harvesting parameters are pre-programmed to **import 5 Items per demonstration site** for a total of 10 Items. You can customize the harvesting settings according to the instructions in this section: Harvester Configuration

Start the migration from DSPace to VIVO

- make sure Solr and Tomcat are running (see Run - Start/Stop VIVO and Visual confirmation in your web browser)
- Compile ETL-migration program

Run migrating process

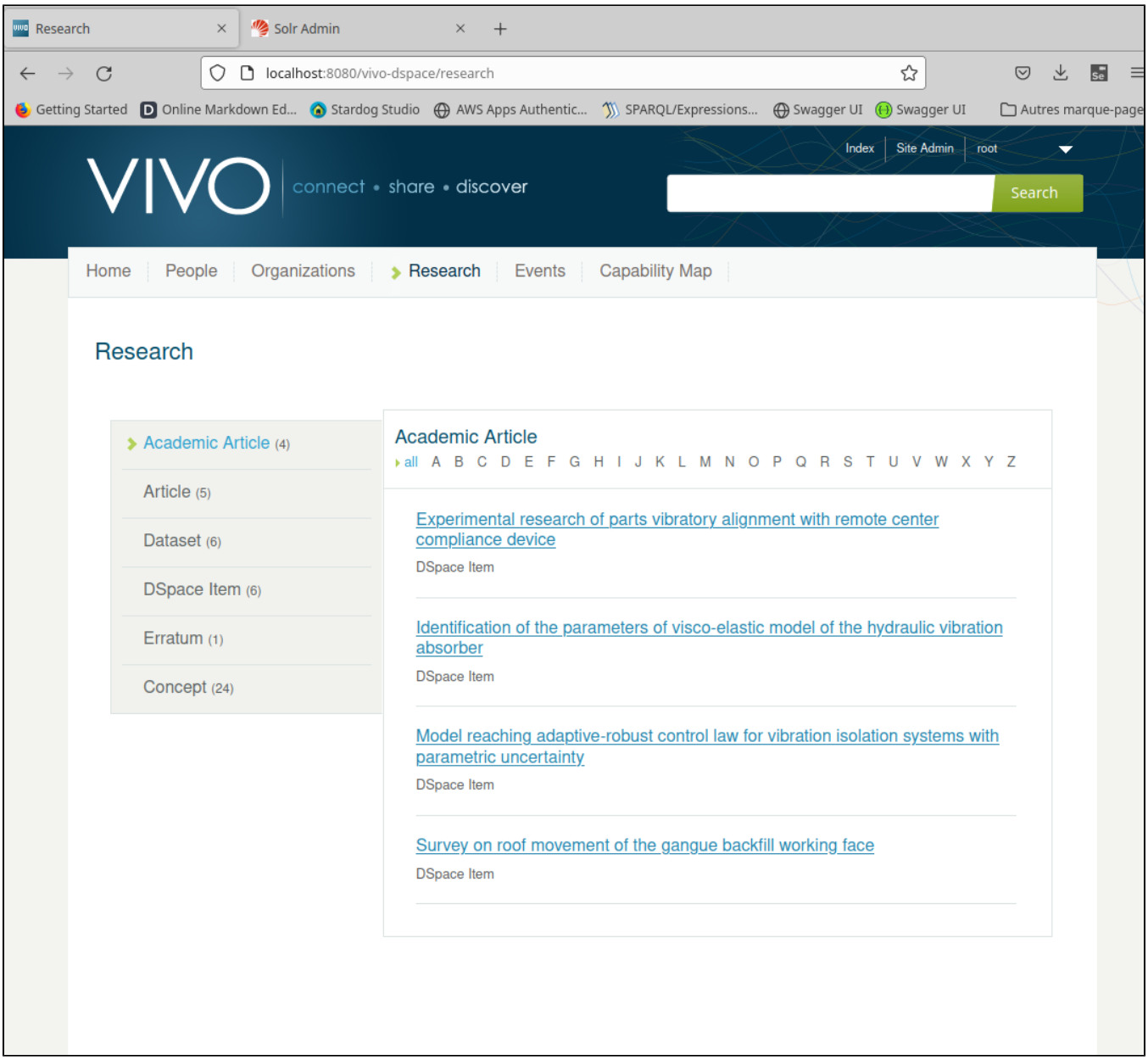
```
./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/script/mvn_install_example.sh
```

- Run ETL-Migration program

Run migrating process

```
./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/script/ETL-migration-DSpace-VIVO.sh
```

Sample result in VIVO after running the migration scripts



Customize the harvesting process

The initial harvesting process is configured to achieve a fast result. The configuration values can be adjusted to meet your experimental needs

Harvester configuration

To change the harvesting parameters of DemoDSpace6 and DemoDspace-7, edit and adjust the configuration values of the following two files:

Config file path to configure Dspace source
`./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/src/main/resources/harvester-dspace6.conf` `./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/src/main/resources/harvester-dspace7.conf`

In the file you will find an example of the fields that the configuration file should contain. At the moment two ways of harvesting the data are supported:

- OAI : It uses the OAI-PMH protocol supported by the Dspace repositories for information harvesting. Make sure you have an OAI-PMH endpoint enabled in the Dspace repository.
- Restv7 : It uses the Dspace APIs to query data. At the moment it is only compatible with versions of Dspace 7.

The following table shows the main fields that are required to configure the information harvesting process from dspace

Params	Description	Example
type	Type of API used for data extraction. Enable for now (OAI, RESTv7)	OAI
endpoint	API url direction. If the OAI type was selected, place the route of the OAI endpoint. On the other hand, if RESTv7 was selected, the rest API address.	https://api7.dspace.org /server/oai/request
uriPrefix	The prefix of the link to the repository. Used to generate valid links to the source repository.	https://demo7.dspace.org/
harvestTotalCount	Number of items to harvest (proposed value = 5); put the parameter in comment to harvest all the site items	5
username	User of the dspace platform with permissions to use the rest API. (RESTv7 Only).	admin
password:	Password of the user (RESTv7 Only).	admin

After the configuration as desired, simply restart the harvesting script as described: Run ETL-Migration program

Technical information

This section aims to present different technical aspects for an extended use of this example

Script directories

The directory `./DSpace-VIVO/test/org.vivoweb.dspacevivo.etlexample/script/` contains bash scripts specific to the execution of the VIVO population scenario from DemoDSpace-6 and DemoDSpace-7. These scripts can inspire the design of scripts needed for harvesting proprietary DSpace instances

Script name nomenclature

The table below presents the list of scripts whose name is built according to the *prefix-function_name*.sh nomenclature

```
00-env.sh                              flush_data_dspace.sh                    func-skip-first-line.
sh                map-document-with-author-to-vivo.sh            produce-list-of-persons.sh
clean-all-transformation-directory.sh  func-capitalize-each-fisrt-caracter.sh   func-sort-list.
sh                      map-expertise-and-item-to-a-person-to-vivo.sh  transformation-map-dc_type.sh
ETL-migration-DSpace-VIVO.sh           func-clean-begin-ending-whitespace.sh    get-vivo-bibo-label.
sh                 map-expertise-to-vivo.sh                       transform-map-expertise-and-item-to-a-person-
to-vivo.sh
extract-dspace6.sh                     func-encode_string_to_expertise.sh       load-data-doc_type-to-vivo.
sh          map-name-to-vivo-person.sh                     transform-map-vivo-doc-type.sh
extract-dspace7.sh                     func-encode_string_to_i18n_lowercase.sh  load-data-expertises-to-vivo.
sh        map-vivo-doc-type.sh                           transform-map-vivo-expertises.sh
extract-dspace.sh                      func-encode_string_to_uid.sh             load-data-person-expertise-to-
vivo.sh  mvn_install_example.sh                          transform-map-vivo-person.sh
flush_data_dspace6.sh                  func-encode_string_to_vivo-URI.sh        load-data-person-to-vivo.
sh            produce-list-of-expertise.sh
flush_data_dspace7.sh                  func-remove-brace-to-uri.sh              load-data-to-vivo.
sh                  produce-list-of-itemtype.sh
```

The table below shows the meaning of the prefixes:

Prefix	Description
extract-	Script prefixes for the data extraction step
transform-	Script prefixes for the data transformation step
load-	Script prefixes for the data loading step
func-	Generic functions
map-	Script for mapping DSpace data to the VIVO vocabulary. These scripts contain the SPARQL constuct queries needed for the mapping

produce-	Production scripts for the various lists needed for ETL processes

Specific scripts

The directory also contains scripts dedicated to specific actions

mvn_install_example.sh

Script used to compile Java programs

00-env.sh

This file is used to define the environment variables needed to run the extract/transform/load (ETL) process of dspace2vivo. Each script includes (source) this file

The code block below shows the list and meaning of the environment variables necessary for proper execution of the scripts

00-env.sh content

```
#!/bin/bash

#####################################################################
# Script Name   : 00-env.sh
# Description   : This file is used to define the environment variables
#                 needed to run the extract/transform/load (ETL)
#                 process of dspace2vivo
# Args          :
# Author        : Michel Héon PhD
# Institution   : Université du Québec à Montréal
# Copyright     : Université du Québec à Montréal (c) 2022
# Email         : heon.michel@uqam.ca
#####################################################################
# Scripts root directory
export LOC_SCRIPT_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" &> /dev/null && pwd -P)"

#####################################################################
# Root installation directory of the different dspace2vivo packages
export INSTALLER_DIR=$(cd $LOC_SCRIPT_DIR/../../../releng/org.vivoweb.dspacevivo.installer ; pwd -P)

#####################################################################
# Project root variables
source $INSTALLER_DIR/00-env.sh

#####################################################################
# Executable and script path needed to run dspace2VIVO
PATH=$LOC_SCRIPT_DIR:$PATH

#####################################################################
# Working directory of scripts
export WORKDIR=$(cd $LOC_SCRIPT_DIR/../; pwd -P)

#####################################################################
# Directory of resources needed to configure the expected operation of the scripts
export RESSOURCESDIR=$(cd $WORKDIR/src/main/resources ; pwd -P)

#####################################################################
# Directory containing the correspondence files between DSpace values and VIVO values
export MAPPING_DATA_DIR=$(cd $RESSOURCESDIR/mapping_data ; pwd -P)

#####################################################################
# Resource directories after compilation. This directory is modified at each compilation (Do not edit)
export RESSOURCES_TARGET_DIR=$(cd $WORKDIR/target/classes ; pwd -P)

#####################################################################
# Directory containing the queries necessary for the execution of SPARQL
export QUERY_DIR=$(cd $RESSOURCESDIR/query ; pwd -P)

#####################################################################
# Repositories containing transient data from the extract/transform/load process
export DATA_DIR=$(cd $WORKDIR/data ; pwd -P)
export DATA_DEMO6_DIR=$(cd $WORKDIR/data_src_dspace6 ; pwd -P)
export DATA_DEMO7_DIR=$(cd $WORKDIR/data_src_dspace7 ; pwd -P)

#####################################################################
# Data transition sub-directories for each step of the ETL process
export ETL_DIR_EXTRACT=$DATA_DIR/extract
export ETL_DIR_TRANSFORM=$DATA_DIR/transform
export ETL_DIR_TRANSFORM_DOC_TYPE=$(cd ${ETL_DIR_TRANSFORM}_doc_type ; pwd -P)
export ETL_DIR_TRANSFORM_PERSON=$(cd ${ETL_DIR_TRANSFORM}_person ; pwd -P)
export ETL_DIR_TRANSFORM_EXPERTISES=$(cd ${ETL_DIR_TRANSFORM}_expertises ; pwd -P)
export ETL_DIR_TRANSFORM_PERSON_EXPERTISES=$(cd ${ETL_DIR_TRANSFORM}_person_expertises ; pwd -P)
```

ETL-migration-DSpace-VIVO.sh

This script encapsulates the functions call allowing the migration of DSpace Demo(6&7) data into VIVO. It is the main script of the ETL process

ETL-migration-DSpace-VIVO

```
#!/bin/bash

#####################################################################
# Script Name   :
# Description   : This script encapsulates the functions call allowing the migration of DSpace Demo(6&7) data
into VIVO
# Args          :
# Author               : Michel Héon PhD
# Institution   : Université du Québec à Montréal
# Copyright     : Université du Québec à Montréal (c) 2022
# Email         : heon.michel@uqam.ca
#####################################################################
export SCRIPT_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" &> /dev/null && pwd -P)"
source $SCRIPT_DIR/00-env.sh
cd $SCRIPT_DIR

#####################################################################
# Clean and setup up data directories and properties
cp $RESSOURCESDIR/*.conf $RESSOURCES_TARGET_DIR
flush_data_dspace.sh 2>/dev/null
flush_data_dspace6.sh 2>/dev/null
flush_data_dspace7.sh 2>/dev/null
#####################################################################
# Extract dspace(6-7) demo data
./extract-dspace6.sh
./extract-dspace7.sh
cp -r $DATA_DEMO6_DIR/* $DATA_DEMO7_DIR/* $DATA_DIR

#####################################################################
# Produce all list
echo run produce-list-of-expertise.sh
produce-list-of-expertise.sh

###########################
echo run produce-list-of-itemtype.sh
produce-list-of-itemtype.sh

###########################
echo run produce-list-of-persons.sh
produce-list-of-persons.sh

#####################################################################
# Process transformation and load to VIVO
load-data-to-vivo.sh
transform-map-vivo-doc-type.sh
load-data-doc_type-to-vivo.sh ; vivo-recomputeIndex.sh &

transform-map-vivo-person.sh
load-data-person-to-vivo.sh ; vivo-recomputeIndex.sh &

transform-map-vivo-expertises.sh
load-data-expertises-to-vivo.sh ; vivo-recomputeIndex.sh &

transform-map-expertise-and-item-to-a-person-to-vivo.sh
load-data-person-expertise-to-vivo.sh ; vivo-recomputeIndex.sh

#####################################################################
# Done ETL Process
echo "Done!"
```

-- End Of Document --