1. updateL8Metadata

Command: python updateL8Metadata.py

This script downloads Landsat 8 metadata and builds a database. The metadata are updated daily by the USGS as scenes are continuously acquired by the satellite. Instead of reconciling the database on a regular basis, the script deletes all records and imports the newly published metadata. For this reason, it’s preferable to run the script daily via the **Task scheduler**.

Out of the 63 parameter values available for each acquired scene, the script imports only 9 parameters (this can be changed). These parameters are:

Sensor type, Collection number, Collection type, Path, Row, Acquisition date, Scene ID, Product ID and Cloud cover over land.

Metadata download page: <https://landsat.usgs.gov/download-entire-collection-metadata>

1. execWorkflow

Display usage: python execWorkflow.py -h

Options: [-c filename] [-wf classname] [-dt DATES] [-d] [-f] [-r] [-v]

-c filename, --config filename (default: params.conf)

-wf classname, --workflow classname (SAGA Workflow classname)

-dt [start date] [end date], --dates [start date] [end date]

-d, --display Displays file naming convention

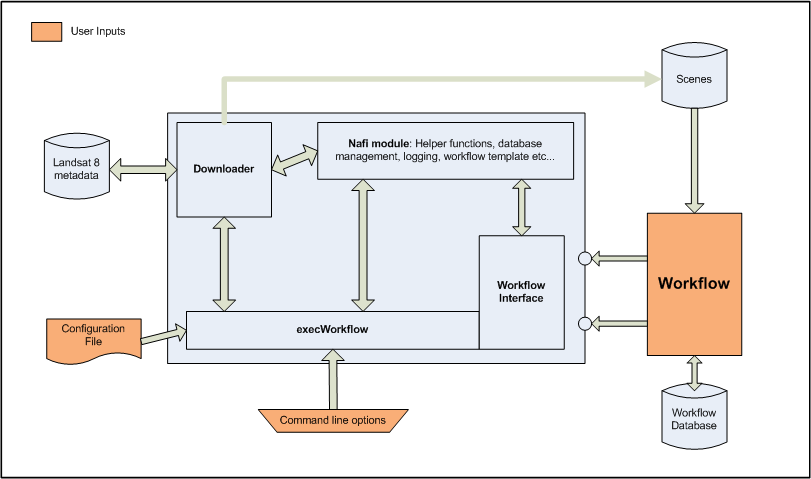
-f, --force Forces run of all workflow computations

-r, --revision Displays script version/revision

-v, --validate Validates configuration file parameters

**execWorkflow.py** is a python programme that downloads Landsat8 scenes and processes them through a workflow. The programme needs the Landsat8 metadata database to work (Figure 1). The programme reads a workflow (external python class) from the command line or from the configuration file and executes it.

The programme is now **stateful**, i.e. it keeps tracks of downloaded files and processing steps. The downloaded files are recorded into the ‘**downloader.db’** database. The workflow processing steps are recorded into a database named after the workflow class name. All these databases can be found under the directory ‘**~\Documents\nafi**’ where ‘**~**’ is the user’s home directory.



**Figure 1**: execWorkflow programme diagram

**Configuration file modifications**:

Section [USGS]:

* I removed the entries (# USGS data repository URLs). This is now done internally, thanks to metadata database queries. Plus, the only active repository for newly acquired OLI/TIRS scenes is the Tier-1 collection level.
* I added the parameter ‘**cc\_land**‘ (cloud coverage over land) It’s in the metadata database anyway, and could be useful down the road. It allows to filter downloads by cloud cover overall percentage. If no value is given all found scenes will be downloaded (assuming up to 100% cloud cover is acceptable)

[LOGGER] new section:

* timestamp: on/off. Add a timestamp to the logfile name
* rotations: Maximum number of logfiles before archiving (rotating logfiles)
* identifier: a string appended to the logfile name. Help to differentiate concurrent scripts execution

Logfile general format: *WorkflowClassname\_(identifier)\_(timestamp)\_N.log*

Parentheses denote an optional field, N is a number between 1 and rotations parameter value.

**Command line options changes**:

* The [-dt] [--dates] option now takes only 2 parameters ‘**begin\_date**‘ and ‘**end\_date**‘. By querying the metadata database, we get all the acquired scenes available within the requested time frame. The step (days) parameter is no longer needed, because we are not trying to download scenes blindly.
* Added the [-wf] [--workflow] option for an external workflow class
* Added the [-f] [--force] option. This will recompute all the workflow processing steps. Basically, it tells the programme to ignore previous computations saved into the database. Useful when SAGA commands have been modified.

1. TestWorkflow

Display usage: python execTestWorkflow.py -h

Options: [-c filename] [-wf classname] [-tf filename] [-d] [-f] [-r] [-v]

-c filename, --config filename (default: params.conf)

-wf classname, --workflow classname (SAGA Workflow classname)

-tf filename, --tarfile filename (Landsat8 scene .tgz file)

-d, --display Displays file naming convention

-f, --force Forces run of all workflow computations

-r, --revision Displays script version/revision

-v, --validate Validates configuration file parameters

A screenshot of a cell phone

Description generated with high confidence

**Figure 2**: TestWorkflow programme diagram

This Python programme is used to develop and test workflows without downloading data. The script takes for inputs a configuration file and a scene tar file. Recognised format for the compressed tar file (.tgz) are either:

Scene ID: LC8PPPRRRYYYYDDDLGN00.tar.gz

or

Scene product ID: LC08\_LLLL\_PPPRRR\_YYYYMMDD\_yyyymmdd\_CC\_TX.tgz

Reference: <https://landsat.usgs.gov/what-are-naming-conventions-landsat-scene-identifiers>

1. LogViewer.py

Display usage: python LogViewer.py -h

Options: [-l logfile] [-p thread name] [-t tag name]

-l logfile, --log logfile

-p thread name, --process thread name

-t tagname, --tags tagname (tag values in INFO, DEBUG, WARNING, CRITICAL, ERROR)

A script used to parse and filter a python logfile. There are two levels of filtering possible. One based on thread name and the other based on python logger tags.

Example:

python LogViewer.py -l logfile -p Downloader -t DEBUG,CRITICAL

This will extract and display ‘Downloader’ thread entries with DEBUG or CRITICAL tags.