RSC Items Transformation

This document explains how “transform\_RSC.py” work, the functions it involves and the dependency between different functions.

Inhalt

[Necessary files or folders 2](#_Toc524941717)

[transform\_RSC.py 2](#_Toc524941718)

[tempjson.json 2](#_Toc524941719)

[pyExcelReader.py 2](#_Toc524941720)

[from\_DOI(f\_doi, doi) 2](#_Toc524941721)

[pyExlDict(filexl) 2](#_Toc524941722)

[urlRequest.py 2](#_Toc524941723)

[loginRequest(namePass): 2](#_Toc524941724)

[affRequest(name, ouID\_MPI): 2](#_Toc524941725)

[upfileRequest(Token, filePath, filename): 3](#_Toc524941726)

[itemsRequest(Token, jsonfile): 3](#_Toc524941727)

[subsidiary\_docs (folder that contains all the .xlsx or .csv mapping files) 3](#_Toc524941728)

[Abbr-RSC.xlsx 3](#_Toc524941729)

[Month.xlsx 3](#_Toc524941730)

[instId\_ctxId.xlsx 3](#_Toc524941731)

[30759 (folder that contain .xml metadata and .pdf files) 3](#_Toc524941732)

[rsc\_201701-201807.csv 3](#_Toc524941733)

# Necessary files or folders

## transform\_RSC.py

This is the main program doing the automatic transformation.

**Transformation idea**: load .xml metadata into a python dict *xmlDict*, load a json metadata template into a python dict *jsondict* and find the values that *jsondict* needed from *xmlDict*. In this RSC transformation, the keys in xml metadata of needed values are pre-known.

**Uploading idea**: Via REST API, do login, pdf uploading and item uploading automatically.

The code is elaborately explained in ***transform\_RSC.py***. Different function blocks are separated by “====” or “----“.

## tempjson.json

This is a template of the json metadata. This file is loaded in ***transform\_RSC.py*** as a python dict *jsondict.*

## pyExcelReader.py

This file contains two help functions *from\_DOI(f\_doi, doi)* and *pyExlDict(filexl).*

### from\_DOI(f\_doi, doi)

This function finds ctx\_ID and ou\_ID of the institute to which the publication belongs.

f\_doi is the file that contains mapping from doi to the 3 digit MPI\_ID.

doi is the digital object id of the publication.

A subsidiary file “instID\_ctxID.xlsx” is needed in this function. This file maps MPI\_ID to ctx\_ID and ou\_ID. This file should be located in “subsidiary\_doc” folder. This file is not complete yet (13.09.2018) and should be completed as soon as possible.

### pyExlDict(filexl)

This function transform a 2-column excel file into a dict, with first column as key and second column as value. This function is used for month string to digit mapping and RSC journal abbreviation to full name mapping only so far (13.09.2018).

## urlRequest.py

This file contains 4 request functions performing automatic interaction with REST API.

### loginRequest(namePass):

Login in Pure, get the Token which is necessary for further pdf upload and item upload.

namePass should in the form of “name:password”.

### affRequest(name, ouID\_MPI):

Search for the ou\_ID of the name in Pure via REST API.

Name is a name of an affiliation of an author.

ouID\_MPI is the ou\_ID found by doi mapping. Elastic search won’t be done for those MPI internal affiliations when the ouID\_MPI is not “xxx”.

The search is done by elastic search. (For more information: <https://www.elastic.co/guide/en/elasticsearch/reference/current/search-request-body.html>, <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html>) Since the elastic search works very “loose” in order to find at least one match for the name, it may cause some mismatch for those names which are not yet in Pure.

### upfileRequest(Token, filePath, filename):

Upload corresponding pdf files of the item via REST API, get the upfileId, which will be the value of the key ‘content’ of ‘files’ in json format data.

In case there’s no pdf attached, the function will return “No PDF” for the main program to handle.

### itemsRequest(Token, jsonfile):

Upload the metadata jsonfile to Pure via REST API.

## subsidiary\_docs (folder that contains all the .xlsx or .csv mapping files)

This folder contains the LUTs (.xlsx or .csv files) used in ***transform\_RSC.py*** for information mappings.

### Abbr-RSC.xlsx

This file gives mapping from the shortcuts of RSC journals to their full-names.

This file is referred by ***transform\_RSC.py*** as *fileAbbrRSC* (line 43).

### Month.xlsx

This file gives mapping from the string representations of 12 months to their integer representation.

This file is referred by ***transform\_RSC.py*** as *fileMonthNum* (line 42).

### instId\_ctxId.xlsx

This file contains the mappings from 3-digit MPI\_ID (column 0) of MPI institutes to their ctx\_ID(column 1, used for item forwarding) and ou\_ID(column 2, used for affiliation links).

This file is relatively consistent and is referred by ***from\_DOI()*** function.

## 30759 (folder that contain .xml metadata and .pdf files)

This folder contains all the .xml metadata to be transformed, the corresponding pdf to be attached and a summary sheet *rsc\_201701-201807.csv* which is explained below.

### rsc\_201701-201807.csv

This file contains the mapping from the doi (column 0) of a publication item to the 3-digit MPI\_ID (column 3) of the corresponding MPI institute.

This file is referred by ***transform\_RSC.py*** as *fileDOIaff* (line 44).

This file will be updated regularly with the coming .xml metadata.