ReQUIAM_csv

Release 0.12.0

Chun Ly

Mar 15, 2021

CONTENTS:

ReQUIAMcsv | CSV Generation Tool for ReQUIAM

Research themes and organization mapping to work with figshare patron management

CHAPTER

ONE

OVERVIEW

Constructs a mapping list between research themes ("portals") and EDS/LDAP organization code to work with our Figshare patron management software (ReQUIAM). This code will generate a CSV file that is used for automation. The code imports a Google Sheet that is maintained by the Data Repository Team. The advantages of using Google Sheets are:

1. Ease of use (no need to format CSV)
2. Advanced spreadsheet capabilities with `MATCH()`, and permitting/prohibiting cells for modification
3. Documentation capabilities via comments and version history management
4. Ability to grant access to University of Arizona Libraries staff for coordinated maintenance

With the above Google Sheet that is imported as a CSV file using `pandas`, it generates a CSV file called `data/research_themes.csv`. There are two versions of this file:

- Trusted version, `master`: [raw] [rendered]
- Under developement, `develop`: [raw] [rendered]

The *workflow* describes how version control will be conducted with these two different branches. In general, after a maintainer implements a change to the Google Sheet, s/he will perform an update to the `develop` branch. Once that has been reviewed, a pull request will be done to merge the changes into the `master` branch.

CHAPTER

TWO

GETTING STARTED

These instructions will have the code running on your local or virtual machine.

2.1 Requirements

You will need the following to have a working copy of this software. See *installation* steps:

1. Python (>=3.7.9)
2. numpy (1.18.0)
3. pandas (0.25.3)

2.2 Installation Instructions

2.2.1 Python and setting up a `conda` environment

First, install a working version of Python (>=3.7.9). We recommend using the Anaconda package installer.

After you have Anaconda installed, you will want to create a separate `conda` environment and activate it:

```
$ (sudo) conda create -n rsh_themes python=3.7
$ conda activate rsh_themes
```

Next, clone this repository into a parent folder:

```
(rsh_themes) $ cd /path/to/parent/folder
(rsh_themes) $ git clone https://github.com/UAL-ODIS/ReQUIAM_csv.git
```

With the activated `conda` environment, you can install with the `setup.py` script:

```
(rsh_themes) $ cd /path/to/parent/folder/ReQUIAM_csv
(rsh_themes) $ (sudo) python setup.py develop
```

This will automatically installed the required `numpy` and `pandas` packages.

You can confirm installation via `conda list`

```
(rsh_themes) $ conda list requiam_csv
```

You should see that the version is `0.12.0`.

2.3 Configuration Settings

Configuration settings are specified through the default.ini file. These settings include the Google Sheet information and CSV file names (do **not** change as this will break ReQUIAM).

2.4 Testing Installation

To test the installation and create a temporary CSV file that does not affect the main CSV file, the following command will run and generate a file called `dry_run.csv`:

```
(rsh_themes) $ python requiam_csv/script_run
```

CHAPTER

THREE

EXECUTION

By default, the script does a "dry run." To execute the script and override the main CSV file (data/research_themes.csv), include the `execute` argument

```
(rsh_themes) $ python requiam_csv/script_run --execute
```

3.1 Workflow

The recommended workflow to commit changes on the main CSV file is as follows:

1. First, switch to `develop` branch: `git checkout develop`
2. Conduct a dry run execution
3. Compare the two CSV files: 'data/research_themes.csv' and 'data/dry_run.csv'
4. If the changes are what you expect, conduct the full execution
5. Update the version number in README.md, `__init__.py`, and setup.py
6. Perform a `git add` and `git commit` for 'data/research_themes.csv' and the above files to `develop`
7. Create a pull request here
8. Update your local git repository with `git pull --all`

CHAPTER

FOUR

VERSIONING

We use SemVer for versioning. For the versions available, see the tags on this repository.

CHAPTER

FIVE

AUTHORS

- Chun Ly, Ph.D. (@astrochun) - University of Arizona Libraries, Office of Digital Innovation and Stewardship

See also the list of contributors who participated in this project.

CHAPTER

SIX

LICENSE

This project is licensed under the MIT License - see the LICENSE file for details.

CHAPTER

SEVEN

API DOCUMENTATION

7.1 ReQUIAM_csv package

7.1.1 Submodules

commons module

requiam_csv.commons.**no_org_code_index**(*df*)

Identify entries without an Org Code. This is based on whether the value is set to NaN

Parameters **df** (DataFrame) – Research Themes dataframe

Return type ndarray

Returns Array containing elements

create_csv module

requiam_csv.create_csv.**create_csv**(*url*, *outfile*, *log*)

Generates a list of organization codes and associated portals for figshare account management.

- The initial spreadsheet, which is curated by UA Libraries, is provided through the [url] input.
- The exported CSV file will be placed in this git repo. Current path and file preference:

  requiam_csv/data/research_themes.csv

Parameters

- **url** (str) – Full url to CSV
- **outfile** (str) – Exported file in CSV format
- **log** (Logger) – Logger object

inspect_csv module

requiam_csv.inspect_csv.**inspect_csv**(*df*, *log*)

Inspects Google Sheet CSV-export table to identify issues. Minor issues are logged. Major issues prevent creating the final CSV file.

Minor issues include:

- Entries without an 'Org Code' (i.e., empty rows). Minor because it is excluded in final export

Major issues include:

- Duplicate entries based on Org Code
- Invalid/incorrect entries in 'Departments/Colleges/Labs/Centers' This result in not getting a proper Org Code
- Missing 'Research Themes' or Sub-portals if either one is provided

Parameters

- **df** (DataFrame) – Research Themes dataframe
- **log** (Logger) – Logger object

logger module

class requiam_csv.logger.**LogClass**(*log_dir*, *logfile*)

Bases: object

Main class to log information to stdout and ASCII logfile

To use: `log = LogClass(log_dir, logfile).get_logger()`

Parameters

- **log_dir** (str) – Relative path for exported logfile directory
- **logfile** (str) – Filename for exported log file

get_logger()

Return type Logger

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

r

INDEX