github2pandas

Release 1.1.18

Maximilian Karl & Sebastian Zug

Nov 23, 2021

CONTENTS

CHAPTER

ONE

TRANSFORM GITHUB ACTIVITIES TO PANDAS DATAFRAMES

1.1 General information

This package is being developed by the participating partners (TU Bergakademie Freiberg, OVGU Magdeburg and HU Berlin) as part of the DiP-iT project Website.

The package implements Python functions for

- aggregating and preprocessing GitHub activities (Commits, Actions, Issues, Pull-Requests) and
- generating project progress summaries according to different metrics (ratio of changed lines, ratio of aggregated Levenshtein distances e.g.).

`github2pandas` stores the collected information in a collection of pandas DataFrames starting from a user defined root folder. The structure beyond that (file names, folder names) is defined as a member variable in the corresponding classes and can be overwritten. The default configuration results in the following file structure.

```
|-- My_Github_Repository_0                <- Repository name
|   |- Repo.json                          <- Json file containing user and repo name
|   |- Repository
|   |   |- Repository.p
|   |- Issues
|   |   |- pdIssuesComments.p
|   |   |- pdIssuesEvents.p
|   |   |- pdIssues.p
|   |   |- pdIssuesReactions.p
|   |- PullRequests
|   |   |- pdPullRequestsComments.p
|   |   |- pdPullRequestsCommits.p
|   |   |- pdPullRequestsEvents.p
|   |   |- pdPullRequests.p
|   |   |- pdPullRequestsReactions.p
|   |   |- pdPullRequestsReviews.p
|   |- Users.p
|   |- Versions
|   |   |- pdCommits.p
|   |   |- pdEdits.p
|   |   |- pdBranches.p
|   |   |- pVersions.db
|   |   |- repo                         <- Repository clone
|   |   |   |- ..
|   |- Workflows
|       |- pdWorkflows.p
|-- My_Github_Repository_1
...
```

The internal structure and relations of the data frames are included in the project's wiki.

1.2 Installation

`github2pandas` is available on pypi. Use pip to install the package.

1.2.1 global

On Linux:

```
sudo pip3 install github2pandas
sudo pip install github2pandas
```

On Windows as admin or for one user:

```
pip install github2pandas
pip install --user github2pandas
```

1.2.2 in virtual environment:

```
pipenv install github2pandas
```

1.3 Usage

GitHub token is required for use, which is used for authentication. The website describes how you can generate this for your GitHub account. Customise the username and project name and explore any public or private repository you have access to with your account!

Access token is to define in `.env` oder `.py (.ipynb)` file. The default value of python.envFile setting is `${workspaceFolder}/.env`

```
TOKEN="example_token"
```

An short example of a python script:

```
import os

from github2pandas.issues import Issues
from github2pandas.utility import Utility
from pathlib import Path

git_repo_name = "github2pandas"
git_repo_owner = "TUBAF-IFI-DiPiT"

default_data_folder = Path("data", git_repo_name)
github_token = os.environ['TOKEN']

repo = Utility.get_repo(git_repo_owner, git_repo_name, github_token, default_data_
↪folder)
Issues.generate_issue_pandas_tables(repo, default_data_folder)
```

(continues on next page)

(continued from previous page)

```
issues = Issues.get_issues(default_data_folder,Issues.ISSUES)

# List the last 14 issue entries
issues.head(14)
```

1.4 Notebook examples

The corresponding github2pandas_notebooks repository illustrates the usage with examplary investigations.

The documentation of the module is available at https://github2pandas.readthedocs.io/.

1.5 Working with pipenv

Process	Command
Installation	`pipenv install --dev`
Run specific script	`pipenv run python file.py`
Run all Tests	`pipenv run python -m unittest`
Run all tests in a specific folder	`pipenv run python -m unittest discover -s 'tests'`
Run all tests with specific filename	`pipenv run python -m unittest discover -p 'test_*.py'`
Start Jupyter server in virtual environment	`pipenv run jupyter notebook`

CHAPTER

TWO

FOR CONTRIBUTORS

Naming conventions: https://namingconvention.org/python/

CHAPTER

THREE

GITHUB2PANDAS PACKAGE

3.1 Submodules

3.2 github2pandas.git_releases module

class github2pandas.git_releases.**GitReleases**

Bases: object

Class to aggregate git releases.

GIT_RELEASES_DIR

Git releases dir where all files are saved in.

Type str

GIT_RELEASES

Pandas table file for git releases data.

Type str

extract_git_releases_data(*git_release*, *users_ids*, *data_root_dir*)

Extracting general git release data.

generate_git_releases_pandas_tables(*repo*, *data_root_dir*, *check_for_updates=True*)

Extracting the complete git releases data from a repository.

get_git_releases(*data_root_dir*, *filename=GIT_RELEASES*)

Get a genearted pandas table.

GIT_RELEASES = 'pdReleases.p'

GIT_RELEASES_DIR = 'Releases'

static extract_git_releases_data(*git_release*, *users_ids*, *data_root_dir*)

Extracting general git release data.

Parameters

- **git_release** (*GitRelease*) – GitRelease object from pygithub.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Dictionary with the extracted general git release data.

Return type dict

Notes

PyGithub GitRelease object structure: https://pygithub.readthedocs.io/en/latest/github_objects/GitRelease.html

static generate_git_releases_pandas_tables(*repo*, *data_root_dir*, *check_for_updates=True*)

Extracting the complete git releases data from a repository.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **check_for_updates** (*bool, default=True*) – Check first if there are any new git releases information.

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_git_releases(*data_root_dir*, *filename=GIT_RELEASES*)

Get a genearted pandas table.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **filename** (*str, default=GIT_RELEASES*) – Pandas table file for git releases data

Returns Pandas DataFrame which can includes the desired data

Return type DataFrame

3.3 github2pandas.issues module

class github2pandas.issues.**Issues**

Bases: object

Class to aggregate Issues

ISSUES_DIR

Issues dir where all files are saved in.

Type str

ISSUES

Pandas table file for issues data.

Type str

ISSUES_COMMENTS

Pandas table file for comments data in issues.

Type str

ISSUES_REACTIONS

Pandas table file for reactions data in issues.

Type str

ISSUES_EVENTS
Pandas table file for reviews data in issues.

Type str

extract_issue_data(*issue*, *users_ids*, *data_root_dir*)
Extracting general issue data.

generate_issue_pandas_tables(*repo*, *data_root_dir*, *reactions=False*, *check_for_updates=True*)
Extracting the complete issue data from a repository.

get_issues(*data_root_dir*, *filename=ISSUES*)
Get a genearted pandas table.

ISSUES = 'pdIssues.p'

ISSUES_COMMENTS = 'pdIssuesComments.p'

ISSUES_DIR = 'Issues'

ISSUES_EVENTS = 'pdIssuesEvents.p'

ISSUES_REACTIONS = 'pdIssuesReactions.p'

static extract_issue_data(*issue*, *users_ids*, *data_root_dir*)
Extracting general issue data.

Parameters

- **issue** (*Issue*) – Issue object from pygithub.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Dictionary with the extracted general issue data.

Return type dict

Notes

PyGithub Issue object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Issue.html

static generate_issue_pandas_tables(*repo*, *data_root_dir*, *reactions=False*, *check_for_updates=True*)
Extracting the complete issue data from a repository.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **reactions** (*bool, default=False*) – If reactions should also be exracted. The extraction of all reactions increases significantly the aggregation speed.
- **check_for_updates** (*bool, default=True*) – Check first if there are any new issues information.

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_issues(*data_root_dir*, *filename=ISSUES*)
Get a genearted pandas table.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **filename** (*str, default=ISSUES*) – Pandas table file for issues or comments or reactions or events data.

Returns Pandas DataFrame which can include the desired data

Return type DataFrame

3.4 github2pandas.pull_requests module

class github2pandas.pull_requests.**PullRequests**
Bases: object

Class to aggregate Pull Requests

PULL_REQUESTS_DIR
Pull request dir where all files are saved in.

Type str

PULL_REQUESTS
Pandas table file for pull request data.

Type str

PULL_REQUESTS_COMMENTS
Pandas table file for comments data in pull requests.

Type str

PULL_REQUESTS_REACTIONS
Pandas table file for reactions data in pull requests.

Type str

PULL_REQUESTS_REVIEWS
Pandas table file for reviews data in pull requests.

Type str

PULL_REQUESTS_EVENTS
Pandas table file for events data in pull requests.

Type str

PULL_REQUESTS_COMMITS
Pandas table file for commits data in pull requests.

Type str

extract_pull_request_data(*pull_request*, *users_ids*, *data_root_dir*)
Extracting general pull request data.

extract_pull_request_review_data(*review*, *pull_request_id*, *users_ids*, *data_root_dir*)
Extracting general review data from a pull request.

extract_pull_request_commit_data(*review*, *users_ids*, *pull_request_id*)
Extracting commit data from a pull request.

generate_pull_request_pandas_tables(*repo*, *data_root_dir*, *reactions=False*, *check_for_updates=True*)
Extracting the complete pull request data from a repository.

get_pull_requests(*data_root_dir*, *filename=PULL_REQUESTS)*)
Get a genearted pandas table.

PULL_REQUESTS = 'pdPullRequests.p'

PULL_REQUESTS_COMMENTS = 'pdPullRequestsComments.p'

PULL_REQUESTS_COMMITS = 'pdPullRequestsCommits.p'

PULL_REQUESTS_DIR = 'PullRequests'

PULL_REQUESTS_EVENTS = 'pdPullRequestsEvents.p'

PULL_REQUESTS_REACTIONS = 'pdPullRequestsReactions.p'

PULL_REQUESTS_REVIEWS = 'pdPullRequestsReviews.p'

static extract_pull_request_commit_data(*review*, *users_ids*, *pull_request_id*)
Extracting commit data from a pull request.

Parameters

- **commit** (*Commit*) – Commit object from pygithub.
- **pull_request_id** (*int*) – Pull request id as foreign key.

Returns Dictionary with the extracted commit data.

Return type dict

Notes

PyGithub Commit object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Commit.html

static extract_pull_request_data(*pull_request*, *users_ids*, *data_root_dir*)
Extracting general pull request data.

Parameters

- **pull_request** (*PullRequest*) – PullRequest object from pygithub.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Dictionary with the extracted general pull request data.

Return type dict

Notes

PyGithub PullRequest object structure: https://pygithub.readthedocs.io/en/latest/github_objects/PullRequest.html

static extract_pull_request_review_data(*review*, *users_ids*, *pull_request_id*)
Extracting review data from a pull request.

Parameters

- **review** (*PullRequestReview*) – PullRequestReview object from pygithub.
- **pull_request_id** (*int*) – Pull request id as foreign key.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Dictionary with the extracted review data.

Return type dict

Notes

PyGithub PullRequestReview object structure: https://pygithub.readthedocs.io/en/latest/github_objects/PullRequestReview.html

static generate_pull_request_pandas_tables(*repo*, *data_root_dir*, *reactions=False*, *check_for_updates=True*)
Extracting the complete pull request data from a repository.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **reactions** (*bool, default=False*) – If reactions should also be exracted. The extraction of all reactions increases significantly the aggregation speed.
- **check_for_updates** (*bool, default=True*) – Check first if there are any new pull requests information.

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_pull_requests(*data_root_dir*, *filename=PULL_REQUESTS)*)
Get a genearted pandas table.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **filename** (*str, default=PULL_REQUESTS*) – Pandas table file for pull requests or comments or reactions or reviews or events data.

Returns Pandas DataFrame which can includes the desired data

Return type DataFrame

3.5 github2pandas.utility module

class github2pandas.utility.**Utility**
Bases: object

Class which contains methods for mutiple modules.

USERS
Pandas table file for user data.

Type str

REPO
Json file for general repository informations.

Type str

check_for_updates(*new_list*, *old_df*)
Check if id and updated_at are in the old_df.

check_for_updates_paginated(*new_paginated_list*, *old_df*)
Check if id and updated_at are in the old_df.

save_list_to_pandas_table(*dir*, *file*, *data_list*)
Save a data list to a pandas table.

get_repo_informations(*data_root_dir*)
Get a repository data (owner and name).

get_repos(*token*, *data_root_dir*, *whitelist_patterns=None*, *blacklist_patterns=None*)
Get mutiple repositorys by pattern and token.

get_repo(*repo_owner*, *repo_name*, *token*, *data_root_dir*)
Get a repository by owner, name and token.

apply_datetime_format(*pd_table*, *source_column*, *destination_column=None*)
Provide equal date formate for all timestamps.

get_users(*data_root_dir*)
Get the generated users pandas table.

get_users_ids(*data_root_dir*)
Get the generated useres as dict whith github ids as keys and anonym uuids as values.

extract_assignees(*github_assignees*, *users_ids*, *data_root_dir*)
Get all assignees as one string.

extract_labels(*github_labels*)
Get all labels as one string.

extract_user_data(*user*, *users_ids*, *data_root_dir*, *node_id_to_anonym_uuid=False*)
Extracting general user data.

extract_author_data_from_commit(*repo*, *sha*, *users_ids*, *data_root_dir*)
Extracting general author data from a commit.

extract_committer_data_from_commit(*repo*, *sha*, *users_ids*, *data_root_dir*)
Extracting general committer data from a commit.

extract_reaction_data(*reaction*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general reaction data.

extract_event_data(*event*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general event data from a issue or pull request.

extract_comment_data(*comment*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general comment data from a pull request or issue.

define_unknown_user(*unknown_user_name*, *uuid*, *data_root_dir*, *new_user=False*)
Defines a unknown user. Add unknown user to alias or creates new user

REPO = 'Repo.json'

USERS = 'Users.p'

static apply_datetime_format(*pd_table*, *source_column*, *destination_column=None*)
Provide equal date formate for all timestamps

Parameters

- **pd_table** (*pandas Dataframe*) – List of NamedUser
- **source_column** (*str*) – Source column name.
- **destination_column** (*str, default=None*) – Destination column name. Saves to Source if None.

Returns String which contains all assignees.

Return type str

static check_for_updates(*new_list*, *old_df*)
Check if id and updated_at are in the old_df.

Parameters

- **new_list** (*list*) – new list with id and updated_at.
- **old_df** (*DataFrame*) – old Dataframe.

Returns True if the repo needs to be updated. False the List is uptodate.

Return type bool

static check_for_updates_paginated(*new_paginated_list*, *old_df*)
Check if id and updated_at are in the old_df.

Parameters

- **new_paginated_list** (*PaginatedList*) – new paginated list with id and updated_at.
- **old_df** (*DataFrame*) – old Dataframe.

Returns True if it need to be updated. False the List is uptodate.

Return type bool

static define_unknown_user(*unknown_user_name*, *uuid*, *data_root_dir*, *new_user=False*)
Defines a unknown user. Add unknown user to alias or creates new user

Parameters

- **unknown_user_name** (*str*) – Name of unknown user.
- **uuid** (*str*) – Uuid can be the anonym uuid of another user or random uuid for a new user.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **new_user** (*bool, default=False*) – A complete new user with anonym_uuid will be generated.

Returns Uuid of the user.

Return type str

static extract_assignees(*github_assignees*, *users_ids*, *data_root_dir*)
Get all assignees as one string.

Parameters

- **github_assignees** (*list*) – List of NamedUser.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns String which contains all assignees and are connected with the char &.

Return type str

Notes

PyGithub NamedUser object structure: https://pygithub.readthedocs.io/en/latest/github_objects/NamedUser.html

static extract_author_data_from_commit(*repo*, *sha*, *users_ids*, *data_root_dir*)
Extracting general author data from a commit.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **sha** (*str*) – sha from the commit.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Anonym uuid of user.

Return type str

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static extract_comment_data(*comment*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general comment data from a pull request or issue.

Parameters

- **comment** (*github_object*) – PullRequestComment or IssueComment object from pygithub.
- **parent_id** (*int*) – Id from parent as foreign key.
- **parent_name** (*str*) – Name of the parent.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Repo dir of the project.

Returns Dictionary with the extracted data.

Return type CommentData

Notes

PullRequestComment object structure: https://pygithub.readthedocs.io/en/latest/github_objects/PullRequestComment.html IssueComment object structure: https://pygithub.readthedocs.io/en/latest/github_objects/IssueComment.html

static extract_committer_data_from_commit(*repo*, *sha*, *users_ids*, *data_root_dir*)
Extracting general committer data from a commit.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **sha** (*str*) – sha from the commit.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Anonym uuid of user.

Return type str

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static extract_event_data(*event*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general event data from a issue or pull request.

Parameters

- **t** (*even*) – IssueEvent object from pygithub.
- **parent_id** (*int*) – Id from parent as foreign key.
- **parent_name** (*str*) – Name of the parent.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Repo dir of the project.

Returns Dictionary with the extracted data.

Return type EventData

Notes

IssueEvent object structure: https://pygithub.readthedocs.io/en/latest/github_objects/IssueEvent.html

static extract_labels(*github_labels*)
Get all labels as one string.

Parameters **github_labels** (*list*) – List of Label.

Returns String which contains all labels and are connected with the char &.

Return type str

Notes

PyGithub Label object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Label.html

static extract_reaction_data(*reaction*, *parent_id*, *parent_name*, *users_ids*, *data_root_dir*)
Extracting general reaction data.

Parameters

- **reaction** (*Reaction*) – Reaction object from pygithub.
- **parent_id** (*int*) – Id from parent as foreign key.
- **parent_name** (*str*) – Name of the parent.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Repo dir of the project.

Returns Dictionary with the extracted data.

Return type ReactionData

Notes

Reaction object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Reaction.html

static extract_user_data(*user*, *users_ids*, *data_root_dir*, *node_id_to_anonym_uuid=False*)
Extracting general user data.

Parameters

- **user** (*NamedUser*) – NamedUser object from pygithub.
- **users_ids** (*dict*) – Dict of User Ids as Keys and anonym Ids as Value.
- **data_root_dir** (*str*) – Repo dir of the project.
- **node_id_to_anonym_uuid** (*bool, default=False*) – Node_id will be the anonym_uuid

Returns Anonym uuid of user.

Return type str

Notes

PyGithub NamedUser object structure: https://pygithub.readthedocs.io/en/latest/github_objects/NamedUser.html

static get_repo(*repo_owner*, *repo_name*, *token*, *data_root_dir*)
Get a repository by owner, name and token.

Parameters

- **repo_owner** (*str*) – the owner of the desired repository.
- **repo_name** (*str*) – the name of the desired repository.
- **token** (*str*) – A valid Github Token.
- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Repository object from pygithub.

Return type repo

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_repo_informations(*data_root_dir*)
Get a repository data (owner and name).

Parameters **data_root_dir** (*str*) – Data root directory for the repository.

Returns Repository Owner and name

Return type tuple

static get_repos(*token*, *data_root_dir*, *whitelist_patterns=None*, *blacklist_patterns=None*)
Get mutiple repositorys by mutiple pattern and token.

Parameters

- **token** (*str*) – A valid Github Token.
- **data_root_dir** (*str*) – Data root directory for the repositorys.
- **whitelist_patterns** (*list*) – the whitelist pattern of the desired repository.
- **blacklist_patterns** (*list*) – the blacklist pattern of the desired repository.

Returns List of Repository objects from pygithub.

Return type List

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_users(*data_root_dir*)
Get the generated users pandas table.

Parameters **data_root_dir** (*str*) – Data root directory for the repository.

Returns Pandas DataFrame which includes the users data

Return type DataFrame

static get_users_ids(*data_root_dir*)
Get the generated useres as dict whith github ids as keys and anonym uuids as values.

Parameters **data_root_dir** (*str*) – Data root directory for the repository.

Returns Dict whith github ids as keys and anonym uuids as values.

Return type dict

static save_list_to_pandas_table(*dir*, *file*, *data_list*)
Save a data list to a pandas table.

Parameters

- **dir** (*str*) – Path to the desired save dir.
- **file** (*str*) – Name of the file.

- **data_list** (*list*) – list of data dictionarys

3.6 github2pandas.version module

class github2pandas.version.**Version**
Bases: object

Class to aggregate Version

VERSION_DIR
Version dir where all files are saved in.

Type str

VERSION_REPOSITORY_DIR
Folder of cloned repository.

Type str

VERSION_COMMITS
Pandas table file for commits.

Type str

VERSION_EDITS
Pandas table file for edit data per commit.

Type str

VERSION_BRANCHES
Pandas table file for branch names.

Type str

VERSION_DB
MYSQL data base file containing version history.

Type str

no_of_processes
Number of processors used for crawling process.

Type int

COMMIT_DELETEABLE_COLUMNS
Commit colums from git2net which can be deleted.

Type list

COMMIT_RENAMING_COLUMNS
Commit Colums from git2net which need to be renamed.

Type dict

EDIT_RENAMING_COLUMNS
Edit Colums from git2net which need to be renamed.

Type dict

handleError(*func*, *path*, *exc_info*)
Error handler function which will try to change file permission and call the calling function again.

clone_repository(repo, data_root_dir, github_token=None, new_clone=False):
Cloning repository from git.

generate_data_base(*data_root_dir*)
Extracting version data from a local repository and storing them in a mysql data base.

generate_version_pandas_tables(*repo*, *data_root_dir*, *check_for_updates=True*)
Extracting edits and commits in a pandas table.

define_unknown_user(*unknown_user_name*, *uuid*, *data_root_dir*, *new_user=False*)
Define unknown user in commits pandas table.

get_unknown_users(*data_root_dir*)
Get all unknown users in from commits.

get_version(*data_root_dir*, *filename=VERSION_COMMITS*)
Get the generated pandas table.

COMMIT_DELETEABLE_COLUMNS = ['author_email', 'author_name', 'committer_email', 'author

COMMIT_RENAMING_COLUMNS = {'committer_date': 'commited_at', 'hash': 'commit_sha', 'p

EDIT_RENAMING_COLUMNS = {'commit_hash': 'commit_sha'}

VERSION_BRANCHES = 'pdBrances.p'

VERSION_COMMITS = 'pdCommits.p'

VERSION_DB = 'Versions.db'

VERSION_DIR = 'Versions'

VERSION_EDITS = 'pdEdits.p'

VERSION_REPOSITORY_DIR = 'repo'

static clone_repository(*repo*, *data_root_dir*, *github_token=None*, *new_clone=False*)
Clone_repository(repo, data_root_dir, github_token=None)

Cloning repository from git.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Repo dir of the project.
- **github_token** (*str*) – Token string.
- **new_clone** (*bool, default=True*) – Initiating a completely new clone of the repository

Notes

Pygit2 documentation: https://github.com/libgit2/pygit2

static define_unknown_user(*unknown_user_name*, *uuid*, *data_root_dir*, *new_user=False*)
Define unknown user in commits pandas table.

Parameters

- **unknown_user_name** (*str*) – Name of unknown user.
- **uuid** (*str*) – Uuid can be the anonym uuid of another user or random uuid for a new user.
- **data_root_dir** (*str*) – Data root directory for the repository.

- **new_user** (*bool, default=False*) – A complete new user with uuid will be generated.

static generate_data_base(*data_root_dir*)
Extracting version data from a local repository and storing them in a mysql data base.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **new_extraction** (*bool, default = False*) – Start a new complete extraction run

Notes

Be aware of the large number of configuration parameters for appling the crawling process given by https://github.com/gotec/git2net/blob/master/git2net/extraction.py

```
def mine_git_repo(git_repo_dir, sqlite_db_file, commits=[],
                  use_blocks=False, no_of_processes=os.cpu_count(), chunksize=1,
↪ exclude=[],
                  blame_C='', blame_w=False, max_modifications=0, timeout=0,␣
↪extract_text=False,
                  extract_complexity=False, extract_merges=True, extract_merge_
↪deletions=False,
                  all_branches=False):
```

static generate_version_pandas_tables(*repo*, *data_root_dir*)
Extracting edits and commits in a pandas table.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **check_for_updates** (*bool, default=True*) – Check first if there are any new pull requests information.

static get_unknown_users(*data_root_dir*)
Get all unknown users in from commits.

Parameters **data_root_dir** (*str*) – Data root directory for the repository.

Returns List of unknown user names

Return type List

static get_version(*data_root_dir*, *filename=VERSION_COMMITS*)
Get the generated pandas table.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **filename** (*str, default=VERSION_COMMITS*) – Pandas table file for commits or edits.

Returns Pandas DataFrame which includes the commit or edit data set

Return type DataFrame

static handleError(*func*, *path*, *exc_info*)
Error handler function which will try to change file permission and call the calling function again.

Parameters

- **func** (*Function*) – Calling function.
- **path** (*str*) – Path of the file which causes the Error.
- **exc_info** (*str*) – Execution information.

no_of_proceses = 1

3.7 github2pandas.workflows module

class github2pandas.workflows.**Workflows**

Bases: object

Class to aggregate Workflows

WORKFLOWS_DIR

workflow dir where all files are saved in.

Type str

WORKFLOWS

Pandas table file for workflow data.

Type str

WORKFLOWS_RUNS

Pandas table file for run data.

Type str

extract_workflow_data(*workflow*)

Extracting general workflow data.

extract_workflow_run_data(*workflow_run*)

Extracting general workflow run data.

generate_workflow_pandas_tables(*repo*, *data_root_dir*, *check_for_updates=True*)

Extracting the complete workflow list and run history from a repository.

download_workflow_log_files(*repo*, *github_token*, *workflow_run_id*, *data_root_dir*)

Receive workflow log files from GitHub.

get_workflows(*data_root_dir*, *filename=WORKFLOWS*)

Get a generated pandas tables.

WORKFLOWS = 'pdWorkflows.p'

WORKFLOWS_DIR = 'Workflows'

WORKFLOWS_RUNS = 'pdWorkflowsRuns.p'

static download_workflow_log_files(*repo*, *github_token*, *workflow_run_id*, *data_root_dir*)

Receive workflow log files from GitHub.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **github_token** (*str*) – Authentication token for GitHub access.
- **workflow_run_id** (*int*) – Workflow Run Id to download one specific workflow run.

- **data_root_dir** (*str*) – Data root directory for the repository.

Returns Number of downloaded files.

Return type int

Notes

Download api https://docs.github.com/en/rest/reference/actions#list-jobs-for-a-workflow-run Generation of python code based on https://curl.trillworks.com/ PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html PyGithub WorkflowRun object structure: https://pygithub.readthedocs.io/en/latest/github_objects/WorkflowRun.html

static extract_workflow_data(*workflow*)

Extracting general workflow data.

Parameters **workflow** (*Workflow*) – Workflow object from pygithub.

Returns Dictionary with the extracted data.

Return type dict

Notes

PyGithub Workflow object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Workflow.html

static extract_workflow_run_data(*workflow_run*)

Extracting general workflow run data.

Parameters **workflow_run** (*WorkflowRun*) – WorkflowRun object from pygithub.

Returns Dictionary with the extracted data.

Return type dict

Notes

PyGithub WorkflowRun object structure: https://pygithub.readthedocs.io/en/latest/github_objects/WorkflowRun.html

static generate_workflow_pandas_tables(*repo*, *data_root_dir*, *check_for_updates=True*)

Extracting the complete workflow list and run history from a repository.

Parameters

- **repo** (*Repository*) – Repository object from pygithub.
- **data_root_dir** (*str*) – Data root directory for the repository.
- **check_for_updates** (*bool, default=True*) – Check first if there are any new workflows or workflow_runs information.

Notes

PyGithub Repository object structure: https://pygithub.readthedocs.io/en/latest/github_objects/Repository.html

static get_workflows(*data_root_dir*, *filename=WORKFLOWS*)
Get a generated pandas tables.

Parameters

- **data_root_dir** (*str*) – Data root directory for the repository.
- **filename** (*str, default=WORKFLOWS*) – Pandas table file for workflows or workflows runs data.

Returns Pandas DataFrame which can include the desired data.

Return type DataFrame

3.8 Module contents

CHAPTER

FOUR

CHANGE LOG

4.1 Version 1.0.1 (April 21, 2021)

- Publish project

4.2 Version 1.0.2 (April 23, 2021)

- Speed improvemetns

4.3 Version 1.0.3 (April 29, 2021)

- documentation improvements
- minor bug fix

4.4 Version 1.0.31 (April 29, 2021)

- readme fix for pypi

4.5 Version 1.1.0 (Mai 3, 2021)

- Add Tag Names to Commits
- Add Author and committer to Commits (the committer was the author before)
- Adapt documentation

4.6 Version 1.1.1 (Mai 19, 2021)

- Add get mutiple repositorys by whitelist and blacklist pattern

4.7 Version 1.1.2 (Mai 20, 2021)

- Fix get mutiple repositorys by whitelist and blacklist pattern

4.8 Version 1.1.3 (Mai 20, 2021)

- Fix extract_user_data.
- User name can cause an unknown Github exception

4.9 Version 1.1.4 (Mai 20, 2021)

- enhance Fix extract_user_data.

4.10 Version 1.1.5 (Mai 27, 2021)

- add commits sha on pull_request
- solve author and committer problem
- add define_unknown_user to Version
- add get unknown_user from commits
- get_repos has now mutiple whitelist and blacklist pattern and are optional now

4.11 Version 1.1.6 (Mai 28, 2021)

- define unknown users takes now a dictionary in with unknown user as key and id as value. If the user is doesnt exists then a new user will be added.

4.12 Version 1.1.7 (Mai 28, 2021)

- Fix extract user. A name is sometimes not set!

4.13 Version 1.1.8 (July 15, 2021)

- Remove Example notebooks
- bugfix from type in version.py

4.14 Version 1.1.9 (July 28, 2021)

- bugfix in extracting user data from commit

4.15 Version 1.1.10 (July 28, 2021)

- hotfix for 1.1.9

4.16 Version 1.1.11 (July 29, 2021)

- change define unknown user in Utility!
- users can now be referenced with uuids from other users or a new user will be created

4.17 Version 1.1.12 (July 29, 2021)

- solved error: check for numpy is nan in Utility

4.18 Version 1.1.13 (July 29, 2021)

- solved error: ignore Alias if already there in Utility(define_unknown_user)

4.19 Version 1.1.14 (July 30, 2021)

- version download will check if there are defined user for unknown user
- comment out some print
- verion checks now if there are updates before downloading

4.20 Version 1.1.15 (July 30, 2021)

- define unknown user in Version works now only for one user
- if a anonym_uuid is known from a different repository for this unknown user then this anonym uuid will be extract_user_data
- The same unknown Author name will be connected to the same anonym_uuid

4.21 Version 1.1.17 (November 11, 2021)

- add output for crashed git pull operatins
- fix empty repositories

4.22 Version 1.1.18 (November 11, 2021)

- change README intructions
- Excrption handling for release count
- replace git pull by generation of a new clone

PYTHON MODULE INDEX

g

INDEX

G

H

I

M

N

P

R

S

U

V

W