
tuulbachs

Dave Smith

Jun 25, 2020

CONTENTS:

1	tuulbash	3
2	tuulecli	5
2.1	cli_color.py	5
3	tuuldevops	7
3.1	pipeline_steps.py	7
3.2	tag_current_version.py	7
3.3	update_version.py	7
4	tuulgit	9
4.1	check_status.py	9
4.2	tag_commit.py	9
5	tuulver	11
5.1	version.py	11
6	tuulyaml	13
6.1	parse.py	13
6.2	update_simple_value.py	13
7	install	15
8	deploy	17
9	internal API	19
9.1	exception.py	19
10	Indices and tables	21
	Python Module Index	23
	Index	25

Low-level software tuuls, organized into drawers.

**CHAPTER
ONE**

TUULBASH

kickpy - A bash script intended to kick a Python script from an environment that doesn't have an established Python environment yet. For example:

```
./kickpy.sh example.py
```

CHAPTER
TWO

TUULCLI

Tuuls for command line interface (CLI).

2.1 `cli_color.py`

Define colors and font styles for use in CLI output

```
class tuulcli.cli_color.CliColor
    Contain the list of colors and font styles
```

CHAPTER
THREE

TUULDEVOPS

Automation tools for common tasks around software development

3.1 pipeline_steps.py

Provide routines for outputting automated pipeline steps consistently

```
tuuldevops.pipeline_steps.major_step(title, description)  
    Output title and description of a major step in the pipeline
```

3.2 tag_current_version.py

Git tag the commit on the current branch with the current version of this software

```
tuuldevops.tag_current_version.tag_product_version(conf_filename)  
    Git tag the commit on the current branch with the version of this software given in conf_filename
```

3.3 update_version.py

Update the version in the tuulbachs-formatted YAML version file

```
tuuldevops.update_version.update_product_version(conf_filename, new_ver)  
    Write a new_ver as the new value for the 'version' key in the conf_filename
```


TUULGIT

An opinionated set of Git tuuls.

4.1 check_status.py

Check the status of the local git working tree

`tuulgit.check_status.has_staged_uncommitted()`

Return a boolean indicating whether the repository has staged, but uncommitted changes

`tuulgit.check_status.has_unstaged_changes()`

Return a boolean indicating whether the working tree has unstaged changes

`tuulgit.check_status.has_untracked_unignored_files()`

Return a boolean indicating whether the working tree has untracked, unignored files

`tuulgit.check_status.is_clean_working_tree(check_if_working_tree=True)`

Return a boolean indicating whether the Git working tree is clean or not

`tuulgit.check_status.is_working_tree()`

Check if this is a git working tree at all

Raises `TuulError` – when the caller attempts to use this function outside of a git working tree

`tuulgit.check_status.repo_toplevel_path()`

Return a string containing the path of the repo's top-level directory

4.2 tag_commit.py

Git tag the commit on the current branch, *only if* the working tree is clean

`tuulgit.tag_commit.tag_current(tag)`

Git tag (annotated) the current commit from a clean working tree

Raises `TuulError` – when the caller attempts to tag an unclean working tree or to use a tag that already exists on the repo

`tuulgit.tag_commit.tag_current_signed(tag)`

Git tag (signed) the current commit from a clean working tree

Raises `TuulError` – when the caller attempts to tag an unclean working tree or to use a tag that already exists on the repo

`tuulgit.tag_commit.tag_delete_local(tag)`

Delete the named Git tag (local only). This function does *not* delete remote tags

Raises `TuuleError` – if the tag delete fails

TUULVER

Parsing tuuls for a tuulbachs-formatted version YAML input file.

5.1 version.py

Utility functions for managing a tuulbachs-formatted version YAML file

`tuulver.version.bump_build(filename)`

Bump the “build” portion of the version from the input YAML filename

`tuulver.version.bump_major(filename)`

Bump the major portion of the version from the input YAML filename

`tuulver.version.bump_minor(filename)`

Bump the minor portion of the version from the input YAML filename

`tuulver.version.bump_patch(filename)`

Bump the patch portion of the version from the input YAML filename

`tuulver.version.bump_pre(filename, prebase='pre')`

Bump the “pre” portion of the version from the input YAML filename

`tuulver.version.create_version_file(filename, product_name)`

Create an initial tuulbachs-formatted version YAML file

`tuulver.version.emit_product_name(filename)`

Return the product name value from the input YAML filename

`tuulver.version.emit_version(filename)`

Return the version value from the input YAML filename

CHAPTER
SIX

TUULYAML

Low level tuuls for interacting with YAML files.

6.1 parse.py

Parse an input YAML file

```
tuulyaml.parse.parse_yaml (filename)  
    Given input path filename, parse YAML file.
```

6.2 update_simple_value.py

Update a simple top-level value in a YAML file

```
tuulyaml.update_simple_value.update_value (inout_path, existing_key, new_value)  
    Update existing_key to new_value in the existing inout_path YAML file.
```

**CHAPTER
SEVEN**

INSTALL

Note that tuulbachs is not yet published at PyPi.

1. Set up and activate a Python virtual environment at the top level of this project
2. `python -m pip install -r requirements.txt`
3. cd to the local auto directory
4. `./install_local.sh`

DEPLOY

Guidance about how to deploy updates to tuulbachs itself

1. Decide which type of [semantic version](#) upgrade this is (major, minor, patch, etc.)
2. From `tuulver/version.py`, use the appropriate `bump_*` function to update the version string in `version.yaml`
3. Follow [*install*](#) guidance
4. Commit changes to Git
5. From `tuuldevops/tag_current_version.py`, use the `tag_product_version` function to properly tag this release
6. Push the Git update (including tags) to this repo's remotes
7. In a temp dir, [download all required packages](#) without installing them, tar and zip these for deployment.
8. In the src dir, run `pyinstaller --add-data/version.yaml:.. --onefile tuul.py`
9. Publish the release (including offline packages tarball and `tuul` executable) on the repo's remote (Github, for instance)

INTERNAL API

This is code intended for use by tuulbachs itself, not external users.

9.1 exception.py

This module contains the set of tuulbachs' exceptions.

`exception tuulbachs.exception.TuulError(msg=None)`

The parent exception from which all other Tuulbachs Python exceptions are derived.

**CHAPTER
TEN**

INDICES AND TABLES

- genindex
- modindex

PYTHON MODULE INDEX

t

tuulbachs.exception, 19
tuulcli.cli_color, 5
tuuldevops.pipeline_steps, 7
tuuldevops.tag_current_version, 7
tuuldevops.update_version, 7
tuulgit.check_status, 9
tuulgit.tag_commit, 9
tuulver.version, 11
tuulyaml.parse, 13
tuulyaml.update_simple_value, 13

INDEX

B

bump_build() (*in module tuulver.version*), 11
bump_major() (*in module tuulver.version*), 11
bump_minor() (*in module tuulver.version*), 11
bump_patch() (*in module tuulver.version*), 11
bump_pre() (*in module tuulver.version*), 11

C

CliColor (*class in tuulcli.cli_color*), 5
create_version_file() (*in module tuulver.version*), 11

E

emit_product_name() (*in module tuulver.version*), 11
emit_version() (*in module tuulver.version*), 11

H

has_staged_uncommitted() (*in module tuulgit.check_status*), 9
has_unstaged_changes() (*in module tuulgit.check_status*), 9
has_untracked_unignored_files() (*in module tuulgit.check_status*), 9

I

is_clean_working_tree() (*in module tuulgit.check_status*), 9
is_working_tree() (*in module tuulgit.check_status*), 9

M

major_step() (*in module tuuldevops.pipeline_steps*), 7
module
 tuulbachs.exception, 19
 tuulcli.cli_color, 5
 tuuldevops.pipeline_steps, 7
 tuuldevops.tag_current_version, 7
 tuuldevops.update_version, 7
 tuulgit.check_status, 9

tuulgit.tag_commit, 9
tuulver.version, 11
tuulyaml.parse, 13
tuulyaml.update_simple_value, 13

P

parse_yaml() (*in module tuulyaml.parse*), 13

R

repo_toplevel_path() (*in module tuulgit.check_status*), 9

T

tag_current() (*in module tuulgit.tag_commit*), 9
tag_current_signed() (*in module tuulgit.tag_commit*), 9
tag_delete_local() (*in module tuulgit.tag_commit*), 9
tag_product_version() (*in module tuuldevops.tag_current_version*), 7
tuulbachs.exception
 module, 19
tuulcli.cli_color
 module, 5
tuuldevops.pipeline_steps
 module, 7
tuuldevops.tag_current_version
 module, 7
tuuldevops.update_version
 module, 7
TuulError, 19
tuulgit.check_status
 module, 9
tuulgit.tag_commit
 module, 9
tuulver.version
 module, 11
tuulyaml.parse
 module, 13
tuulyaml.update_simple_value
 module, 13

U

update_product_version() (*in module tuulde-*
vops.update_version), [7](#)
update_value() (*in module* *tu-*
ulyaml.update_simple_value), [13](#)