tuulbachs

May 31, 2021

Contents:

Automate low-level, repetitive, yet important development tasks related to semantic versioning, distributed version control, configuration parsing, and build pipelines.

CHAPTER 1

Key Features

- Increment semantic version parts for a versioned entity
- Manage the status of a local Git working tree
- Manage Git tags, local and remote, signed and unsigned
- Parse and update basic YAML configuration files

CHAPTER 2

Quickstart

TBD

2.1 tuulbash

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

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

2.2 tuulcli

Tuuls for command line interface (CLI).

Define colors and font styles for use in CLI output

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

2.3 tuuldevops

Automation tuuls for common tasks around software development

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

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

2.4 tuulgit

An opinionated set of Git tuuls.

2.5 tuulver

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

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

2.6 tuulyaml

Low level tuuls for interacting with YAML files.

Parse an input YAML file

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

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.

2.7 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. `python3 -m pip install pip-tools`
3. `pip-compile`
4. `pip-compile requirements-dev.in`
5. `pip-sync requirements.txt requirements-dev.txt`
6. `cd` to the local `auto` directory
7. `./install_local.sh`

2.8 deploy

How to deploy updates to tuulbachs itself.

Prerequisite: User must have already followed *install* guidance at least once in the target environment.

1. Decide which type of semantic version upgrade this is (major, minor, patch, etc.)
2. In `src/`, run `python3 tuul.py --bump {major, minor, patch}`
3. Follow *install* guidance
4. Commit changes to Git
5. In `src/`, run `python3 tuul.py --autotag`
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. On an available machine with the oldest version of GLIBC that you wish to support, 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)

2.9 internal API

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

Project exception class

exception tuulbachs.exception.**TuulError**(*msg=None*)
Class used for exceptions thrown by tuulbachs

2.10 Indices and tables

- genindex
- modindex

Python Module Index

t

Index