

---

**tuulbachs**

**Apr 25, 2020**



---

## Contents:

---

<b>1</b>	<b>tuulbash</b>	<b>3</b>
<b>2</b>	<b>tuuldevops</b>	<b>5</b>
2.1	tag_current_version.py . . . . .	5
2.2	update_version.py . . . . .	5
<b>3</b>	<b>tuulgit</b>	<b>7</b>
3.1	check_clean.py . . . . .	7
3.2	tag_commit.py . . . . .	7
<b>4</b>	<b>tuulpy</b>	<b>9</b>
<b>5</b>	<b>tuulver</b>	<b>11</b>
5.1	version.py . . . . .	11
<b>6</b>	<b>tuulyaml</b>	<b>13</b>
6.1	parse.py . . . . .	13
6.2	update_simple_value.py . . . . .	13
<b>7</b>	<b>install</b>	<b>15</b>
<b>8</b>	<b>deploy</b>	<b>17</b>
<b>9</b>	<b>Indices and tables</b>	<b>19</b>
	<b>Python Module Index</b>	<b>21</b>
	<b>Index</b>	<b>23</b>



Low-level software tuuls, organized into drawers.



# CHAPTER 1

---

tuulbash

---





Automation tools for common tasks around software development

### 2.1 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

### 2.2 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



An opinionated set of Git tuuls.

### 3.1 check\_clean.py

Check the Git working tree

```
tuulgit.check_clean.is_clean_working_tree()
```

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

### 3.2 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 `TuulError`** – if the tag delete fails



## CHAPTER 4

---

tuulpy

---



## 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_pre(filename, prebase='pre')`

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

`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





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 7

---

### install

---

Note that `tuulbachs` is not yet published at PyPi. So these instructions detail a local installation.

1. `pip install . --upgrade`
2. `cd docs`
3. `make html`



Guidance about how to deploy updates to tuulbachs itself

1. Follow *install* guidance
2. Decide which type of *semantic version* upgrade this is (major, minor, patch, etc.)
3. From `tuulver/version.py`, use the appropriate `bump_*` function to update the version string in `version.yaml`
4. Commit the version change to Git
5. Make sure the local working tree is clean
6. From `tuuldevops/tag_current_version.py`, use the `tag_product_version` function to properly tag this release
7. Push the Git update (including tags) to this repo's remotes



## CHAPTER 9

---

### Indices and tables

---

- `genindex`
- `modindex`





### t

- `tuuldevops.tag_current_version`, [5](#)
- `tuuldevops.update_version`, [5](#)
- `tuulgit.check_clean`, [7](#)
- `tuulgit.tag_commit`, [7](#)
- `tuulver.version`, [11](#)
- `tuulyaml.parse`, [13](#)
- `tuulyaml.update_simple_value`, [13](#)



## B

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

## E

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

## I

`is_clean_working_tree()` (*in module `tuulgit.check_clean`*), 7

## P

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

## T

`tag_current()` (*in module `tuulgit.tag_commit`*), 7  
`tag_current_signed()` (*in module `tuulgit.tag_commit`*), 7  
`tag_delete_local()` (*in module `tuulgit.tag_commit`*), 7  
`tag_product_version()` (*in module `tuuldevops.tag_current_version`*), 5  
`tuuldevops.tag_current_version(module)`, 5  
`tuuldevops.update_version(module)`, 5  
`tuulgit.check_clean(module)`, 7  
`tuulgit.tag_commit(module)`, 7  
`tuulver.version(module)`, 11  
`tuulyaml.parse(module)`, 13  
`tuulyaml.update_simple_value(module)`, 13

## U

`update_product_version()` (*in module `tuuldevops.update_version`*), 5  
`update_value()` (*in module `tuulyaml.update_simple_value`*), 13