toml-sort

Release 0.17.0

Samuel Roeca

Feb 27, 2023

TABLE OF CONTENTS

1 Positional Arguments **3**

2 Named Arguments **5**

3 sort **7**

4 comments **9**

5 formatting **11**

6 Modules **13**
6.1 toml_sort.tomlsort . . . 13

7 toml-sort **15**
7.1 Installation . . . 15
7.2 Motivation . . . 15
7.3 Command line usage . . . 16
7.4 Configuration file . . . 17
7.5 Comments . . . 18
7.5.1 Header . . . 18
7.5.2 Footer . . . 18
7.5.3 Inline . . . 18
7.5.4 Block . . . 19
7.5.5 Orphan . . . 19
7.6 Example . . . 19
7.6.1 Unformatted, unsorted input . . . 19
7.6.2 Formatted, sorted output . . . 20
7.7 Local Development . . . 20
7.8 Written by . . . 21

8 Indices and tables **23**

Python Module Index **25**

Index **27**

See below for a the command line interface.

Toml sort: a sorting utility for toml files.

```
usage: toml-sort [-h] [--version] [-o OUTPUT] [-i] [-I] [-a]
                 [--no-sort-tables] [--sort-table-keys] [--sort-inline-tables]
                 [--sort-inline-arrays] [--no-header] [--no-comments]
                 [--no-header-comments] [--no-footer-comments]
                 [--no-inline-comments] [--no-block-comments]
                 [--spaces-before-inline-comment {1,2,3,4}]
                 [--spaces-indent-inline-array {2,4,6,8}]
                 [--trailing-comma-inline-array] [--check]
                 [F [F ...]]
```

CHAPTER

ONE

POSITIONAL ARGUMENTS

F filename(s) to be processed by toml-sort (default: -)

CHAPTER

TWO

NAMED ARGUMENTS

--version	display version information and exit
	Default: False
-o, --output	output filepath (default: ‘-‘)
-i, --in-place	overwrite the original input file with changes
	Default: False
--check	silently check if an original file would be changed by the formatter
	Default: False

CHAPTER

THREE

SORT

change sorting behavior

-I, --ignore-case ignore case when sorting

Default: False

-a, --all sort ALL keys. This implies sort table-keys, inline-tables and inline arrays. (default: only sort non-inline ‘tables and arrays of tables’)

Default: False

--no-sort-tables Disables the default behavior of sorting tables and arrays of tables by their header value. Setting this option will keep the order of tables in the toml file the same.

Default: False

--sort-table-keys Sort the keys in tables and arrays of tables (excluding inline tables and arrays).

Default: False

--sort-inline-tables Sort inline tables.

Default: False

--sort-inline-arrays Sort inline arrays.

Default: False

CHAPTER

FOUR

COMMENTS

exclude comments from output

--no-header Deprecated. See –no-header-comments

Default: False

--no-comments remove all comments. Implies no header, footer, inline, or block comments

Default: False

--no-header-comments remove a document's leading comments

Default: False

--no-footer-comments remove a document's trailing comments

Default: False

--no-inline-comments remove a document's inline comments

Default: False

--no-block-comments remove a document's block comments

Default: False

CHAPTER

FIVE

FORMATTING

options to change output formatting

--spaces-before-inline-comment Possible choices: 1, 2, 3, 4

the number of spaces before an inline comment (default: 1)

Default: 1

--spaces-indent-inline-array Possible choices: 2, 4, 6, 8

the number of spaces to indent a multiline inline array (default: 2)

Default: 2

--trailing-comma-inline-array add trailing comma to the last item in a multiline inline array

Default: False

Examples:

- **Stdin -> Stdout**: cat input.toml | toml-sort
- **Disk -> Disk**: toml-sort -o output.toml input.toml
- **Linting**: toml-sort –check input.toml input2.toml input3.toml
- **Inplace Disk**: toml-sort –in-place input.toml input2.toml

Return codes:

- 0 : success.
- 1 : errors were found

Notes:

- You cannot redirect from a file to itself in Bash. POSIX shells process redirections first, then execute commands. –in-place exists for this reason

CHAPTER

SIX

MODULES

This page documents importable modules and their contents.

6.1 toml_sort.tomlsort

Utility functions and classes to sort toml text.

class toml_sort.tomlsort.**TomlSort**(*input_toml: str*, *comment_config: Optional[toml_sort.tomlsort.CommentConfiguration] = None*, *sort_config: Optional[toml_sort.tomlsort.SortConfiguration] = None*, *format_config: Optional[toml_sort.tomlsort.FormattingConfiguration] = None*)

API to manage sorting toml files.

aot_to_tomlsortitem(*comments: List[tomlkit.items.Comment]*, *key: tomlkit.items.Key*, *value: tomlkit.items.AoT*) → Tuple[List[tomlkit.items.Comment], toml_sort.tomlsort.TomlSortItem]

Turn an AoT into a TomlSortItem, recursing down through its collections and attaching all the comments to the correct items.

array_sort_func(*value: Tuple[tomlkit.items._ArrayItemGroup, Any]*) → str

Sort function that operates on the .value member of an ArrayItemGroup respects the class setting for ignore_case.

body_to_tomlsortitems(*parent: List[Tuple[Optional[tomlkit.items.Key], tomlkit.items.Item]]*) → Tuple[List[toml_sort.tomlsort.TomlSortItem], List[tomlkit.items.Comment]]

Iterate over Container.body, recursing down into sub-containers attaching the comments that are found to the correct TomlSortItem. We need to do this iteration because TomlKit puts comments into end of the collection they appear in, instead of the start of the next collection.

For example: [xyz]

Comment [abc]

TomlKit would place the comment from the example into the [xyz] collection, when we would like it to be attached to the [abc] collection.

So before sorting we have to iterated over the container, correctly attaching the comments, then undo this process once everything is sorted.

static format_key(*key: tomlkit.items.Key*) → tomlkit.items.Key

Format a key, removing any extra whitespace, and making sure that it will be formatted like: key = value with one space on either side of the equal sign.

key_sort_func(*value: toml_sort.tomlsort.TomlSortItem*) → str

Sort function that looks at TomlSortItems keys, respecting the configured value for ignore_case.

sort_array(*array: tomlkit.items.Array*, *indent_depth: int = 0*) → tomlkit.items.Array

Sort and format an inline array item while preserving comments.

sort_inline_table(*item*, *indent_depth: int = 0*)

Sort an inline table, recursing into its items.

sort_item(*item: tomlkit.items.Item*, *indent_depth: int = 0*) → tomlkit.items.Item

Sort an item, recursing down if the item is an inline table or array.

sort_items(*items: Iterable[toml_sort.tomlsort.TomlSortItem]*) →
Iterable[toml_sort.tomlsort.TomlSortItem]

Sort an iterable full of TomlSortItem, making sure the key is correctly formatted and recursing into any sub-items.

sorted() → str

Sort a TOML string.

sorted_children_table(*parent: List[toml_sort.tomlsort.TomlSortItem]*) →
Iterable[toml_sort.tomlsort.TomlSortItem]

Get the sorted children of a table.

static table_previous_item(*parent_table*, *grandparent*)

Finds the previous item that we should attach a comment to, in the case where the previous item is a table.

This take into account that a table may be a super table.

table_to_tomlsortitem(*comments: List[tomlkit.items.Comment]*, *key: tomlkit.items.Key*, *value: tomlkit.items.Table*) → Tuple[List[tomlkit.items.Comment], toml_sort.tomlsort.TomlSortItem]

Turn a table into a TomlSortItem, recursing down through its collections and attaching all the comments to the correct items.

toml_doc_sorted(*original: tomlkit.toml_document.TOMLDocument*) →
tomlkit.toml_document.TOMLDocument

Sort a TOMLDocument.

toml_elements_sorted(*original: TomlSortItem*, *parent: Table | TOMLDocument*) → Item

Returns a sorted item, recursing collections to their base.

write_header_comment(*from_doc_body: List[Tuple[Optional[tomlkit.items.Key], tomlkit.items.Item]]*, *to_doc: tomlkit.toml_document.TOMLDocument*) → None

Write header comment from the FROM doc to the TO doc.

Only writes comments / whitespace from the beginning of a TOML document.

CHAPTER

SEVEN

TOML-SORT

A command line utility to sort and format your toml files.

Read the latest documentation here: https://toml-sort.readthedocs.io/en/latest/

7.1 Installation

```
pip install toml-sort
```

7.2 Motivation

This library sorts TOML files, providing the following features:

- Sort tables and Arrays of Tables (AoT)
- Option to sort non-tables / non-AoT's, or not
- Preserve comments, where possible
- Standardize whitespace and indentation

I wrote this library/application because I couldn't find any "good" sorting utilities for TOML files. Now, I use this as part of my daily workflow. Hopefully it helps you too!

7.3 Command line usage

This project can be used as either a command line utility or a Python library. Read the docs for an overview of its library capabilities. For command line usage, see below:

```
$ toml-sort --help
usage: toml-sort [-h] [--version] [-o OUTPUT] [-i] [-I] [-a] [--no-sort-tables] [--sort-
↪table-keys]
                 [--sort-inline-tables] [--sort-inline-arrays] [--no-header] [--no-
↪comments] [--no-header-comments]
                 [--no-footer-comments] [--no-inline-comments] [--no-block-comments]
                 [--spaces-before-inline-comment {1,2,3,4}] [--spaces-indent-inline-
↪array {2,4,6,8}]
                 [--trailing-comma-inline-array] [--check]
                 [F [F ...]]

Toml sort: a sorting utility for toml files.

positional arguments:
  F                     filename(s) to be processed by toml-sort (default: -)

optional arguments:
  -h, --help            show this help message and exit
  --version             display version information and exit
  -o OUTPUT, --output OUTPUT
                        output filepath (default: '-')
  -i, --in-place        overwrite the original input file with changes
  --check               silently check if an original file would be changed by the␣
↪formatter

sort:
  change sorting behavior

  -I, --ignore-case     ignore case when sorting
  -a, --all             sort ALL keys. This implies sort table-keys, inline-tables and␣
↪inline arrays. (default: only
                        sort non-inline 'tables and arrays of tables')
  --no-sort-tables      Disables the default behavior of sorting tables and arrays of␣
↪tables by their header value.
                        Setting this option will keep the order of tables in the toml␣
↪file the same.
  --sort-table-keys     Sort the keys in tables and arrays of tables (excluding inline␣
↪tables and arrays).
  --sort-inline-tables  Sort inline tables.
  --sort-inline-arrays  Sort inline arrays.

comments:
  exclude comments from output

  --no-header           Deprecated. See --no-header-comments
  --no-comments         remove all comments. Implies no header, footer, inline, or block␣
↪comments
  --no-header-comments  remove a document's leading comments
```

(continues on next page)

(continued from previous page)

```
  --no-footer-comments  remove a document's trailing comments
  --no-inline-comments  remove a document's inline comments
  --no-block-comments   remove a document's block comments

formatting:
  options to change output formatting

  --spaces-before-inline-comment {1,2,3,4}
                        the number of spaces before an inline comment (default: 1)
  --spaces-indent-inline-array {2,4,6,8}
                        the number of spaces to indent a multiline inline array␣
↪(default: 2)
  --trailing-comma-inline-array
                        add trailing comma to the last item in a multiline inline array

Examples:

  - **Stdin -> Stdout**: cat input.toml | toml-sort
  - **Disk -> Disk**: toml-sort -o output.toml input.toml
  - **Linting**: toml-sort --check input.toml input2.toml input3.toml
  - **Inplace Disk**: toml-sort --in-place input.toml input2.toml

Return codes:

  - 0 : success.
  - 1 : errors were found

Notes:

  - You cannot redirect from a file to itself in Bash. POSIX shells process
    redirections first, then execute commands. --in-place exists for this
    reason
```

7.4 Configuration file

toml-sort can also be configured by using the `pyproject.toml` file. If the file exists and has a `tool.tomlsort` section, the configuration is used. If both command line arguments and the configuration are used, the options are merged. In the case of conflicts, the command line option is used.

In short, the names are the same as on the command line (and have the same meaning), but `-` is replaced with `_`. Please note, that only the below options are supported:

```toml
[tool.tomlsort]
all = true
in_place = true
no_comments = true
no_header_comments = true
no_footer_comments = true
no_inline_comments = true
no_block_comments = true
```

(continues on next page)

(continued from previous page)

```
no_sort_tables = true
sort_table_keys = true
sort_inline_tables = true
sort_inline_arrays = true
spaces_before_inline_comment = 2
spaces_indent_inline_array = 4
trailing_comma_inline_array = true
check = true
ignore_case = true
```

7.5 Comments

Due to the free form nature of comments, it is hard to include them in a sort in a generic way that will work for everyone. `toml-sort` deals with four different types of comments. They are all enabled by default, but can be disabled using CLI switches, in which case comments of that type will be removed from the output.

7.5.1 Header

The first comments in a document, that are followed by a blank line, are treated as a header, and will always remain at the top of the document. If there is no blank line, the comment will be treated as a block comment instead.

```
# This is a header
# it can be multiple lines, as long as it is followed with a blank line
# it will always stay at the top of the sorted document

title = "The example"
```

7.5.2 Footer

Any comments at the end of the document, after the last item in the toml, will be treated as a footer, and will always remain at the bottom of the document.

```
title = "The example"

# this is a footer comment
```

7.5.3 Inline

Inline comments are comments that are at the end of a line where the start of the line is a toml item.

```
title = "The example" # This is a inline comment
```

7.5.4 Block

Block comments, are any comments that are on their own line. These comments are treated as *attached* to the item in the toml that is directly below them, not separated by whitespace. These comments can be multiple lines. Inline comments will appear in the sorted output above the item they were attached to in the input toml.

```
# Comment attached to title
title = "The example"

# This comment is an orphan because it
# is separated from a-section by whitespace

# This comment is attached to a-section
# attached comments can be multiple lines
[a-section]
# This comment is attached to date
date = "2019"
```

7.5.5 Orphan

Orphan comments are any comments that don't fall into the above categories, they will be removed from the output document.

```
# Header comment

# Orphan comment, not attached to any item
# because there is whitespace before title

title = "The example"

# This comment is an orphan because it
# is separated from a-section by whitespace

# This comment is attached to a-section
[a-section]
```

7.6 Example

The following example shows the input, and output, from the CLI with default options.

7.6.1 Unformatted, unsorted input

```
# My great TOML example

  title = "The example"

[[a-section.hello]]
ports = [ 8001, 8001, 8002 ]
dob = 1979-05-27T07:32:00Z # First class dates? Why not?
```

(continues on next page)

(continued from previous page)

```
    # Attached to b-section
  [b-section]
  date = "2018"
  name = "Richard Stallman"

[[a-section.hello]]
ports = [ 80 ]
    #Attached to dob
dob = 1920-05-27T07:32:00Z # Another date!

                          [a-section]
                          date = "2019"
                          name = "Samuel Roeca"
```

7.6.2 Formatted, sorted output

```
# My great TOML example

title = "The example"

[a-section]
date = "2019"
name = "Samuel Roeca"

[[a-section.hello]]
ports = [ 8001, 8001, 8002 ]
dob = 1979-05-27T07:32:00Z # First class dates? Why not?

[[a-section.hello]]
ports = [ 80 ]
# Attached to dob
dob = 1920-05-27T07:32:00Z # Another date!

# Attached to b-section
[b-section]
date = "2018"
name = "Richard Stallman"
```

7.7 Local Development

Local development for this project is quite simple.

Dependencies

Install the following tools manually.

- Poetry>=1.0
- GNU Make

Recommended

- asdf

Set up development environment

```
make setup
```

Run Tests

```
make test
```

7.8 Written by

Samuel Roeca, *samuel.roeca@gmail.com*

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex

PYTHON MODULE INDEX

t

toml_sort.tomlsort, 13

INDEX

A

aot_to_tomlsortitem() (*toml_sort.tomlsort.TomlSort method*), 13
array_sort_func() (*toml_sort.tomlsort.TomlSort method*), 13

B

body_to_tomlsortitems() (*toml_sort.tomlsort.TomlSort method*), 13

F

format_key() (*toml_sort.tomlsort.TomlSort static method*), 13

K

key_sort_func() (*toml_sort.tomlsort.TomlSort method*), 13

M

module
    toml_sort.tomlsort, 13

S

sort_array() (*toml_sort.tomlsort.TomlSort method*), 14
sort_inline_table() (*toml_sort.tomlsort.TomlSort method*), 14
sort_item() (*toml_sort.tomlsort.TomlSort method*), 14
sort_items() (*toml_sort.tomlsort.TomlSort method*), 14
sorted() (*toml_sort.tomlsort.TomlSort method*), 14
sorted_children_table() (*toml_sort.tomlsort.TomlSort method*), 14

T

table_previous_item() (*toml_sort.tomlsort.TomlSort static method*), 14
table_to_tomlsortitem() (*toml_sort.tomlsort.TomlSort method*), 14
toml_doc_sorted() (*toml_sort.tomlsort.TomlSort method*), 14
toml_elements_sorted() (*toml_sort.tomlsort.TomlSort method*), 14
toml_sort.tomlsort
    module, 13
TomlSort (*class in toml_sort.tomlsort*), 13

W

write_header_comment() (*toml_sort.tomlsort.TomlSort method*), 14