yamllint

Release 1.8.1

Oct 04, 2017

Contents

1 Screenshot **3**

2 Table of contents **5**
- 2.1 Quickstart . . . 5
- 2.2 Configuration . . . 6
- 2.3 Rules . . . 10
- 2.4 Disable with comments . . . 12
- 2.5 Development . . . 13
- 2.6 Integration with text editors . . . 13

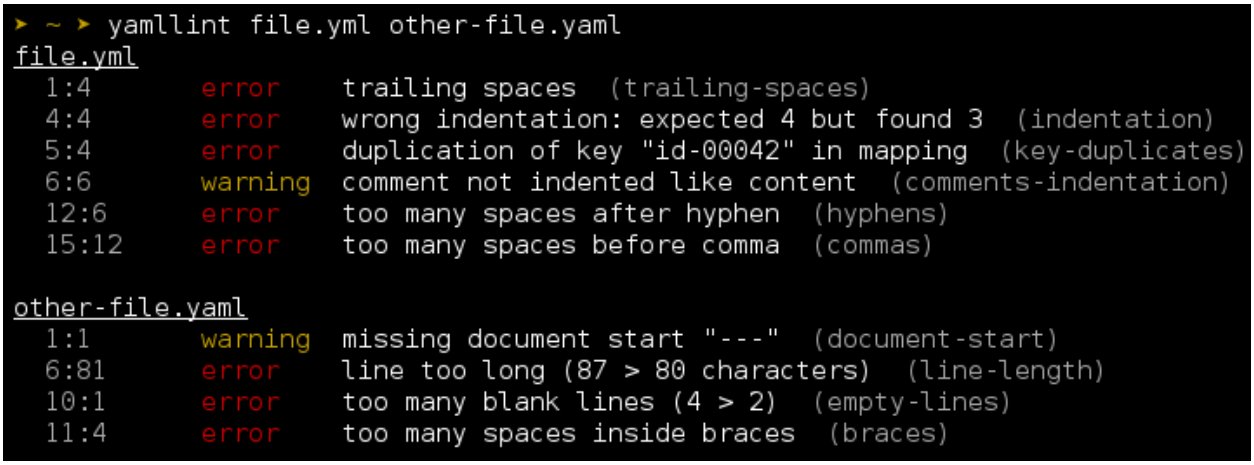
Python Module Index **15**

A linter for YAML files.

yamllint does not only check for syntax validity, but for weirdnesses like key repetition and cosmetic problems such as lines length, trailing spaces, indentation, etc.

CHAPTER 1

Screenshot



Note: The default output format is inspired by eslint, a great linting tool for Javascript.

CHAPTER 2

Table of contents

Quickstart

Installing yamllint

On Fedora / CentOS:

```
sudo dnf install yamllint
```

On Debian 8+ / Ubuntu 16.04+:

```
sudo apt-get install yamllint
```

On older Debian / Ubuntu versions:

```
sudo add-apt-repository -y ppa:adrienverge/ppa && sudo apt-get update
sudo apt-get install yamllint
```

Alternatively using pip, the Python package manager:

```
sudo pip install yamllint
```

If you prefer installing from source, you can run, from the source directory:

```
python setup.py sdist
sudo pip install dist/yamllint-*.tar.gz
```

Running yamllint

Basic usage:

```
yamllint file.yml other-file.yaml
```

You can also lint all YAML files in a whole directory:

```
yamllint .
```

The output will look like (colors are not displayed here):

```
file.yml
  1:4       error    trailing spaces  (trailing-spaces)
  4:4       error    wrong indentation: expected 4 but found 3  (indentation)
  5:4       error    duplication of key "id-00042" in mapping  (key-duplicates)
  6:6       warning  comment not indented like content  (comments-indentation)
  12:6      error    too many spaces after hyphen  (hyphens)
  15:12     error    too many spaces before comma  (commas)

other-file.yaml
  1:1       warning  missing document start "---"  (document-start)
  6:81      error    line too long (87 > 80 characters)  (line-length)
  10:1      error    too many blank lines (4 > 2)  (empty-lines)
  11:4      error    too many spaces inside braces  (braces)
```

Add the `-f parsable` arguments if you need an output format parsable by a machine (for instance for *syntax highlighting in text editors*). The output will then look like:

```
file.yml:6:2: [warning] missing starting space in comment (comments)
file.yml:57:1: [error] trailing spaces (trailing-spaces)
file.yml:60:3: [error] wrong indentation: expected 4 but found 2 (indentation)
```

If you have a custom linting configuration file (see *how to configure yamllint*), it can be passed to yamllint using the `-c` option:

```
yamllint -c ~/myconfig file.yaml
```

Note: If you have a `.yamllint` file in your working directory, it will be automatically loaded as configuration by yamllint.

Configuration

yamllint uses a set of *rules* to check source files for problems. Each rule is independent from the others, and can be enabled, disabled or tweaked. All these settings can be gathered in a configuration file.

To use a custom configuration file, use the `-c` option:

```
yamllint -c /path/to/myconfig file-to-lint.yaml
```

If `-c` is not provided, yamllint will look for a configuration file in the following locations (by order of preference):

- `.yamllint` in the current working directory
- `$XDG_CONFIG_HOME/yamllint/config`
- `~/.config/yamllint/config`

Finally if no config file is found, the default configuration is applied.

Default configuration

Unless told otherwise, yamllint uses its `default` configuration:

```
---

rules:
  braces:
    min-spaces-inside: 0
    max-spaces-inside: 0
    min-spaces-inside-empty: -1
    max-spaces-inside-empty: -1
  brackets:
    min-spaces-inside: 0
    max-spaces-inside: 0
    min-spaces-inside-empty: -1
    max-spaces-inside-empty: -1
  colons:
    max-spaces-before: 0
    max-spaces-after: 1
  commas:
    max-spaces-before: 0
    min-spaces-after: 1
    max-spaces-after: 1
  comments:
    level: warning
    require-starting-space: true
    min-spaces-from-content: 2
  comments-indentation:
    level: warning
  document-end: disable
  document-start:
    level: warning
    present: true
  empty-lines:
    max: 2
    max-start: 0
    max-end: 0
  hyphens:
    max-spaces-after: 1
  indentation:
    spaces: consistent
    indent-sequences: true
    check-multi-line-strings: false
  key-duplicates: enable
  line-length:
    max: 80
    allow-non-breakable-words: true
    allow-non-breakable-inline-mappings: false
  new-line-at-end-of-file: enable
  new-lines:
    type: unix
  trailing-spaces: enable
  truthy:
    level: warning
```

Details on rules can be found on *the rules page*.

There is another pre-defined configuration named `relaxed`. As its name suggests, it is more tolerant:

```
---

extends: default

rules:
  braces:
    level: warning
    max-spaces-inside: 1
  brackets:
    level: warning
    max-spaces-inside: 1
  colons:
    level: warning
  commas:
    level: warning
  comments: disable
  comments-indentation: disable
  document-start: disable
  empty-lines:
    level: warning
  hyphens:
    level: warning
  indentation:
    level: warning
    indent-sequences: consistent
  line-length:
    level: warning
    allow-non-breakable-inline-mappings: true
  truthy: disable
```

It can be chosen using:

```
yamllint -d relaxed file.yml
```

Extending the default configuration

When writing a custom configuration file, you don't need to redefine every rule. Just extend the `default` configuration (or any already-existing configuration file).

For instance, if you just want to disable the `comments-indentation` rule, your file could look like this:

```
# This is my first, very own configuration file for yamllint!
# It extends the default conf by adjusting some options.

extends: default

rules:
  comments-indentation: disable  # don't bother me with this rule
```

Similarly, if you want to set the `line-length` rule as a warning and be less strict on block sequences indentation:

```
extends: default

rules:
  # 80 chars should be enough, but don't fail if a line is longer
  line-length:
```

```
    max: 80
    level: warning

  # accept both      key:
  #                    - item
  #
  # and              key:
  #                  - item
  indentation:
    indent-sequences: whatever
```

Custom configuration without a config file

It is possible – although not recommended – to pass custom configuration options to yamllint with the `-d` (short for `--config-data`) option.

Its content can either be the name of a pre-defined conf (example: `default` or `relaxed`) or a serialized YAML object describing the configuration.

For instance:

```
yamllint -d "{extends: relaxed, rules: {line-length: {max: 120}}}" file.yaml
```

Errors and warnings

Problems detected by yamllint can be raised either as errors or as warnings. The CLI will output them (with different colors when using the `standard` output format).

By default the script will exit with a return code `1` *only when* there is one or more error(s).

However if strict mode is enabled with the `-s` (or `--strict`) option, the return code will be:

- `0` if no errors or warnings occur
- `1` if one or more errors occur
- `2` if no errors occur, but one or more warnings occur

Ignoring paths

It is possible to exclude specific files or directories, so that the linter doesn't process them.

You can either totally ignore files (they won't be looked at):

```
extends: default

ignore: |
  /this/specific/file.yaml
  /all/this/directory/
  *.template.yaml
```

or ignore paths only for specific rules:

```
extends: default

rules:
  trailing-spaces:
    ignore: |
      /this-file-has-trailing-spaces-but-it-is-OK.yaml
      /generated/*.yaml
```

Note that this `.gitignore`-style path pattern allows complex path exclusion/inclusion, see the pathspec README file for more details. Here is a more complex example:

```
# For all rules
ignore: |
  *.dont-lint-me.yaml
  /bin/
  !/bin/*.lint-me-anyway.yaml

extends: default

rules:
  key-duplicates:
    ignore: |
      generated
      *.template.yaml
  trailing-spaces:
    ignore: |
      *.ignore-trailing-spaces.yaml
      /ascii-art/*
```

Rules

When linting a document with yamllint, a series of rules (such as `line-length`, `trailing-spaces`, etc.) are checked against.

A *configuration file* can be used to enable or disable these rules, to set their level (*error* or *warning*), but also to tweak their options.

This page describes the rules and their options.

List of rules

- *braces*
- *brackets*
- *colons*
- *commas*
- *comments*
- *comments-indentation*
- *document-end*
- *document-start*

- *empty-lines*
- *hyphens*
- *indentation*
- *key-duplicates*
- *line-length*
- *new-line-at-end-of-file*
- *new-lines*
- *trailing-spaces*
- *truthy*

braces

brackets

colons

commas

comments

comments-indentation

document-end

document-start

empty-lines

hyphens

indentation

key-duplicates

line-length

new-line-at-end-of-file

new-lines

trailing-spaces

truthy

Disable with comments

Disabling checks for a specific line

To prevent yamllint from reporting problems for a specific line, add a directive comment (`# yamllint disable-line ...`) on that line, or on the line above. For instance:

```
# The following mapping contains the same key twice,
# but I know what I'm doing:
key: value 1
key: value 2  # yamllint disable-line rule:key-duplicates

- This line is waaaaaaaaaay too long but yamllint will not report anything about it.
↪# yamllint disable-line rule:line-length
  This line will be checked by yamllint.
```

or:

```
# The following mapping contains the same key twice,
# but I know what I'm doing:
key: value 1
# yamllint disable-line rule:key-duplicates
key: value 2

# yamllint disable-line rule:line-length
- This line is waaaaaaaaaay too long but yamllint will not report anything about it.
  This line will be checked by yamllint.
```

It is possible, although not recommend, to disabled **all** rules for a specific line:

```
# yamllint disable-line
-  {    all : rules ,are disabled   for this line}
```

If you need to disable multiple rules, it is allowed to chain rules like this: `# yamllint disable-line rule:hyphens rule:commas rule:indentation`.

Disabling checks for all (or part of) the file

To prevent yamllint from reporting problems for the whole file, or for a block of lines within the file, use `# yamllint disable ...` and `# yamllint enable ...` directive comments. For instance:

```
# yamllint disable rule:colons
- Lorem       : ipsum
  dolor       : sit amet,
  consectetur : adipiscing elit
# yamllint enable rule:colons

- rest of the document...
```

It is possible, although not recommend, to disabled **all** rules:

```
# yamllint disable
- Lorem       :
        ipsum:
          dolor : [   sit,amet]
-         consectetur : adipiscing elit
# yamllint enable
```

If you need to disable multiple rules, it is allowed to chain rules like this: `# yamllint disable rule:hyphens rule:commas rule:indentation`.

Development

yamllint provides both a script and a Python module. The latter can be used to write your own linting tools:

Integration with text editors

Most text editors support syntax checking and highlighting, to visually report syntax errors and warnings to the user. yamllint can be used to syntax-check YAML source, but a bit of configuration is required depending on your favorite

text editor.

Vim

Assuming that the ALE plugin is installed, yamllint is supported by default. It is automatically enabled when editing YAML files.

If you instead use the syntastic plugin, add this to your `.vimrc`:

```
let g:syntastic_yaml_checkers = ['yamllint']
```

Neovim

Assuming that the neomake plugin is installed, yamllint is supported by default. It is automatically enabled when editing YAML files.

Emacs

If you are flycheck user, you can use flycheck-yamllint integration.

Other text editors

Help wanted!

Your favorite text editor is not listed here? Help us improve by adding a section (by opening a pull-request or issue on GitHub).

Python Module Index

y

yamllint, 3

Index

Y

yamllint (module), 1