yaml4rst Documentation

Release 0.1.5

Robin Schneider

Mar 16, 2017

Contents

Contents:

yaml4rst introduction

yaml4rst is a linting/checking/reformatting tool for YAML files documented with inline RST which goes hand in hand with yaml2rst.

It has been written to help with keeping the `defaults/main.yml` file in Ansible roles of DebOps up-to-date and to assist with writing or including new roles. DebOps uses Sphinx to generate Ansible role documentation which also includes the default role variables. Refer to debops/docs for details.

Usage

The typical use case for this program is to improve the defaults YAML file of Ansible roles.

The recommended way to do this is to commit all your changes in the repository of the role, then run:

```
yaml4rst -e 'ansible_full_role_name=ROLE_OWNER.ROLE_NAME' defaults/main.yml -i
```

from the root of the role repository. Be sure to replace `ROLE_OWNER.ROLE_NAME` with the particular Ansible role name.

This will check and reformat the `defaults/main.yml` file in place.

Now you can check the reformatted file with a diffing/editing tool of your choosing and fix any warning which `yaml4rst` might have emitted.

Refer to `input_files` and `output_files` in the tests directory for automatically tested examples of input and output files.

Note that for continues usage yaml4rst is invoked from debops-optimize when yaml4rst is installed so you might want to try debops-optimize.

Features

Checks for:

- Reasonable variable namespace
- Undocumented variables

Automatically fixes:

- RST sections which are not folds
- Undocumented variables (adds a FIXME for the user)
- Documented variables which are not folds
- YAML documents without a defined header
- Spacing between variables and sections

Known limitations

- Does not handle folds with implicit level and missing closing fold marker.

  Status: Should be doable but currently not needed nor implemented. A `NotImplementedError` exception is thrown which causes the CLI program to terminate immediately with an error and reference to this section.

  As workaround just strip out the opening folds with your favorite editor as `yaml4rst` will add missing folds for sections and variables anyway. Refer to the Makefile (`prepare-real-data` target) where such a workaround is used for integration testing. Note that this is not perfect as can be seen on the `debops. apt_install` test case.

Repository

- GitHub (primary repo with issue tracker)

Documentation

- Read the Docs

Authors

- Robin Schneider

License

GNU Affero General Public License v3 (AGPL-3.0)

Installation

Latest release

You can install `yaml4rst` by invoking the following commands:

```
gpg --recv-keys 'C505 B5C9 3B0D B3D3 38A1  B600 5FE9 2C12 EE88 E1F0'
mkdir --parent /tmp/yaml4rst && cd /tmp/yaml4rst
wget -r -nd -l 1 https://pypi.python.org/pypi/yaml4rst --accept-regex '^https://
↪(test)?pypi\.python\.org/packages/.*\.whl.*'
current_release="$(find . -type f -name '*.whl' | sort | tail -n 1)"
gpg --verify "${current_release}.asc" && pip3 install "${current_release}"
```

Refer to Verifying PyPI and Conda Packages for more details. Note that this might pull down dependencies in an unauthenticated way! You might want to install the dependencies yourself beforehand.

Or if you feel lazy and agree that pip/issues/1035 should be fixed you can also install `yaml4rst` like this:

```
pip3 install yaml4rst
```

Development version

If you want to be more on the bleeding edge of `yaml4rst` development consider cloning the `git` repository and installing from it:

```
gpg --recv-keys 'EF96 BC32 AC57 CFC7 2DF0  1D8C 489A 4D5E C353 C98A'
git clone https://github.com/ypid/yaml4rst.git
cd yaml4rst && git verify-commit HEAD
echo 'Check if the HEAD commit has a good signature and only proceed in that case!' &&
↪ read -r fnord
echo 'Then chose one of the commands below to install yaml4rst and its dependencies:'
pip3 install .
./setup.py develop --user
```

```
./setup.py install --user
./setup.py install
```

CLI interface

Linting/reformatting tool for YAML files documented with inline RST

```
usage: yaml4rst [-h] [-V] [-d] [-v] [-q] [-n]
                [-o OUTPUT_FILE [OUTPUT_FILE ...]] [-i] [-p PRESET]
                [-e CONFIG_KV]
                input_file [input_file ...]
```

Required Arguments

       input_file   One or more file paths to be processed. '-' will read from STDIN.

Optional Arguments

       -V="==SUPPRESS==", --version="==SUPPRESS==" show program's version number and exit

       -d, --debug   Write debugging and higher to STDOUT|STDERR.

       -v, --verbose   Write information and higher to STDOUT|STDERR.

       -q, --quiet, --silent   Only write errors and higher to STDOUT|STDERR.

       -n=True, --no-warn-summary=True If multiple warnings occur in quiet mode, a summary error message is emitted. This switch suppresses it.

       -o=['-'], --output-file=['-'] One or more file paths where the output should be written to. Defaults to '-' which will write the output from a single input file to STDOUT. If multiple input files where given, and –in-place is not used then the number of output files must match the number of input files.

       -i=False, --in-place=False Edit the input file(s) in place.

       -p="debops/ansible", --preset="debops/ansible" Which preset to use. Default: "debops/ansible".

       -e, --config-kv   Additional configuration, refer to the docs of yaml4rst for details.

Configuration

ansible_full_role_name

> Ansible role name. Example: `debops.apt`.
>
> No default.

closing_fold_format_spec

> Format string in str.format syntax used defining how closing folds should look like.
>
> Defaults to:

```
{:>72}
```

wanted_empty_lines_between_items

> Number of empty lines wanted between sections and/or variables. Note that the closing folds are by default space indented and also count as "empty lines" for that matter.
>
> Defaults to:

```
2
```

add_string_for_missing_comment

> The string to add when a undocumented variable/section was found.
>
> Defaults to:

```
FIXME(yaml4rst): Describe what this variable is doing.
```

Design principles

- Idempotent.

  The program can be run against it's output and should not make any changes to it. This property is checked by integration testing.

  This allows users to integrate the program into their workflow and let **yaml4rst** do the repetitive tasks.

- Keep it simple.

  The Python implementation is thought to be close to the way a human would solve the problem. It basically just parses each line of input into fold sections and checks if something needs to be changed. At the end, the sections are formated back to lines and written to the output.

- Committed to excellence.

  This program is reasonably small and it's scope can be clearly defined so that it is possible even for one person to try to follow most of the best practices in Python programming which includes:

  - Extensive unit and integration testing

  - 100 % unit test code coverage (including branch coverage)

  - Python version test matrix

  - Python linting tools like **pylint**

  - Documentation build testing

  - No redundancy

  Furthermore, most of the best practices listed above are automatically checked (where possible) on each commit and before PRs get accepted.

- Why not do it manually? Hoped to be helpful.

  "How long can you work on making a routine task more efficient before you're spending more time than you save?" (Ref: xkcd 1205)

  "I spend a lot of time on this task. I should write a program automating it!" (Ref: xkcd 1205)

Based on the analysis of git-hours it took ypid 118 hours for the initial public commit. This equals 14.75 days (assuming 8 hours a day). The time for ongoing maintenance of the program is not included and can be estimated based on the public commits.

From that it is clear that for ypid, it took much longer to automate the task then it would have been to do it manually. But that is only part of the story because ypid is not the only person doing this kind of maintenance. So it is believed that this effort will still pay of in saving time for everyone doing this and lead to more productivity and higher quality. (Ref: Collective)

- No formal parsing/lexing.

Existing parsers for YAML where found to be unsuited for the use in a statical analyser/reformatter like this program.

Also, ypid is more familiar with this hacky approach then he would like to confess (ref: opening_hours.js).

- Don't reimplement error checking from Ansible or RST parsers.

The input is not extensively checked. Use Ansible and RST before and/or after the reformatting through this program.

CHAPTER 6

yaml4rst package

Submodules

yaml4rst.cli module

Command line interface of yaml4rst

yaml4rst.cli.**main**()

yaml4rst.reformatter module

Reformatting class of yaml4rst

exception yaml4rst.reformatter.**YamlRstReformatterError**

Bases: exceptions.Exception

Exception which is thrown by YamlRstReformatter when a unrecoverable error occurred.

class yaml4rst.reformatter.**YamlRstReformatter**(*preset='debops/ansible'*, *template_path='/home/docs/checkouts/readthedocs.org/user_builds/yaml* *config=None*)

Bases: object

YAML+RST linting/reformatting class with the following features:

Checks for:

•Reasonable variable namespace

•Undocumented variables

Automatically fixes:

•RST sections which are not folds

- Undocumented variables (adds a FIXME for the user)

- Documented variables which are not folds

- YAML documents without a defined header

- Spacing between variables and sections

get_content ()
  Return one string containing all lines.

read_file (*input_file*)
  Read the given input file path and save its content for later processing.

reformat ()
  Process (check/lint/reformat) the instance lines.

write_file (*output_file*)
  Write the instance lines to the given output file path and save its content for later processing.

yaml4rst.defaults module

Default configuration of yaml4rst

yaml4rst.helpers module

Helper functions of yaml4rst

yaml4rst.helpers.**get_first_match** (*pattern, strings, ind_offset=None, limit_pattern=None, break_pattern=None, match=True*)

yaml4rst.helpers.**get_last_index** (*input_list, search_elems, fallback=None*)

yaml4rst.helpers.**get_last_match** (*pattern, strings, ind_offset=None, limit_pattern=None, match=True*)

yaml4rst.helpers.**insert_list** (*base_list, ind, add_list*)

yaml4rst.helpers.**list_index** (*input_list, elem, fallback=None*)

yaml4rst.helpers.**strip_list** (*input_list*)

Module contents

Linting/reformatting Python package for YAML files documented with inline RST

Contributing and issue reporting

You can contribute and report issues in the usual way as documented by GitHub.

If you found a security vulnerability that might put users at risk please send your report/patch to ypid@riseup.net. Please consider using OpenPGP to encrypt your email.

Changelog

This project adheres to Semantic Versioning and human-readable changelog.

yaml4rst master - unreleased

yaml4rst v0.1.5 - 2017-03-16

Fixed

- Fix RST section level detection for folded sections. [ypid]

- End fold started by an unfolded RST section when a new RST section with the same section level begins instead of including the following RST section in the fold. [ypid]

yaml4rst v0.1.4 - 2017-02-21

Security

- The default `yaml.load` method from PyYAML which is used to validate the input YAML file is unsafe. As a result `yaml4rst` would have executed arbitrary code given in the YAML input file.

  Refer to the issue Make load safe_load. This has been fixed by switching to `yaml.safe_load`. [ypid]

yaml4rst v0.1.3 - 2017-02-14

Changed

- Make the FIXME note which yaml4rst adds for missing variable comments more precise. [ypid]

Fixed

- Fix YAML block detection which lead to wrong indention of closing folds. [ypid]

yaml4rst v0.1.2 - 2016-11-18

Fixed

- Python packaging which previously included and installed the unit tests as separate "tests" Python package. Now the built distribution release (Wheel) only contains the actual Python package "yaml4rst" without the unit tests. Checkout the source distribution or the git repository for hacking on the project. Thanks very much to ganto for reporting and providing a patch! [ypid]

yaml4rst v0.1.1 - 2016-11-05

Fixed

- Fix legacy reST section detection when heading is directly follows by a comment. [ypid]
- Fix YAML block detection which lead to wrong indention of closing folds. [ypid]

yaml4rst v0.1.0 - 2016-10-28

Added

- Initial coding and design. [ypid]

Python Module Index

y