FREE eBook

LEARNING

yaml

Free unaffiliated eBook created from
Stack Overflow contributors.

#yaml

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: yaml

It is an unofficial and free yaml ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official yaml.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with yaml

Remarks

YAML is a recursive acronym for "YAML Ain't Markup Language". It is a human readable data serialization standard for all programming languages.

Versions

Version	Release Date
1.0	2004-01-29
1.1	2005-01-18
1.2	2009-10-01

Examples

Basic Yaml syntax

YAML is a text based format allowing to store structured data in a hierarchy. YAML is designed to be human and machine readable with a minimum of overhead. The YAML specification can be found at yaml.org. There is also a reference card

Comments start with `#` and go till newline, comments must be separated from other tokens by whitespace. Whitespace isn't free, indentation must be spaces, not tabs. YAML will consider that lines prefixed with more spaces than the parent key are contained inside it. Moreover, all lines must be prefixed with the same amount of spaces to belong to the same map.

YAML has sequences and mappings as collection types, both can be represented in flow and block style.

An sequence of scalar strings in YAML looks like:

```
[ one, two, three ]   # flow style

# or block style

- one
- two
- three
```

A mapping consists of key/value pairs:

```
index: 4  # block style
```

```
name: nali

# or

{ index: 4, name: nali }   # flow style

# or nested (equivalent of { level: { one: { two: fun } } }):

level:

  one:

    two: fun
```

Basic YAML Types

```
integer: 25
string: "25"
float: 25.0
boolean: true
null type: null
```

YAML Sequential Data

Same list level:

```
- Cat
- Dog
- Goldfish
```

Nested List:

```
-
 - Cat
 - Dog
 - Goldfish
```

Comments

```
# This comment occupies a whole line
- some item # This comment succeeds content of a line
- http://example.com/#nocomment
- "This # does not introduce a comment."
- |
    This is a block scalar.
    A # inside it does not introduce a comment.
   # unless it is less indented than the first line (this is one)
```

Note that for a `#` to introduce a comment, it must either

- occur at the beginning of a line, or
- be preceded by whitespace.

`#` must always be followed by whitespace. `#` inside quoted scalars never start comments. `#` may introduce comments at the end of block scalars, but therefore, it must be less indented than the block scalar's base indentation (which is usually determined by the indentation of its first non-empty line).

Block Style Mappings

With implicit keys:

```
key: value
another key:
  - some
  - more
  - values
[1, 2, 3]: last value, which has a flow style key
```

With implicit and explicit keys:

```
? key
: value
another key:
  - some
  - more
  - values
? [1, 2, 3]
: last value, which has a flow style key
```

`key`, `another key` and `[1, 2, 3]` are keys of the same mapping, although they use different key styles.

Nested mappings:

```
first level:
  second level:
    ? third level
    :
      forth level: value of implicit key
    ? third level, second key
    : value of explicit key
  ?
    mapping as: key of
    another: mapping
  : scalar value of mapping key
first level, second key:
  last value
```

Splitting text strings over multiple lines

```
- Without quotes:
   You can just
   split a long piece of text like this.
```

```
- With quotes:
    "[But be careful:
     if you \"need\" punctuation, put double quotes around it. You can ev\
     en split without spaces by using backslashes."
- Or single quotes:
    'This works
     but isn''t as flexible'
- If you want to keep those new line characters: |
    Then do
    it this way with
    a pipe (|) character. (This string has three \n characters)
- Or you can have just the one final new line: >
    This string has
    just one \n character, at the very end.
- Block indicators:
    Look up >-, >+, |- and |+ for fine tuning.
```

Escaping Characters

YAML supports three styles of escape notation:

1. Entity Escapes

    a. space: " "

    b. colon: ":"

    c. ampersand: "&"

2. Unicode Escapes

    a. space: "\u0020"

    b. single quote: "\u0027"

    c. double quote: "\u0022"

3. Quoted Escapes

    a. double quote in single quote: 'Is "I always lie" a true statement?'

    b. nested double quote: " She said, "I quit" "

    c. nested single quote: ' He was speechless: " '

Read Getting started with yaml online: https://riptutorial.com/yaml/topic/3181/getting-started-with-yaml

Chapter 2: Using anchors and aliases for transcluded content

Examples

Creating an "Array of Dictionaries" table with YAML anchors as row identifiers

```
---
person_table:
  - &person001
    fname:  homer
    lname:  simpson
    role:   dad
    age:    33

  - &person002
    fname:  marge
    lname:  simpson
    role:   mom
    age:    34

  - &person003
    fname:  peter
    lname:  griffin
    role:   dad
    age:    34
```

Problem

- developer wishes to express a table structure in YAML, where each row is referenced by a compact row identifier

Solution

- use YAML anchors, by assigning an anchor identifier to each row in the table
- in YAML, reusable "transclusion identifiers" are called anchors and aliases
- in YAML, reusable "transclusion identifiers" consist of alphanumeric tokens preceeded by an ampersand or asterisk

Rationale

- YAML anchors and aliases allow for increased data normalization
- YAML anchors and aliases enforce DRY (Don't repeat yourself)
- in this example, a table structure can be designed and preserved which closely coincides with a database

Pitfalls

- YAML anchors must be declared before they can be referenced by aliases
- YAML anchors must be unique throughout the document
- failure to specify unique anchors will cause an error on `yaml.load()`
- not all YAML parsers reliably support anchors and aliases

See also

Stackoverflow YAML

Using YAML aliases to cross-reference rows from a YAML table

```
---
person_table:
  - &person001
    fname:  homer
    lname:  simpson
    role:   dad
    age:    33

  - &person002
    fname:  marge
    lname:  simpson
    role:   mom
    age:    34

  - &person003
    fname:  peter
    lname:  griffin
    role:   dad
    age:    34

motto_table:
  - &motto001
    person:   *person001
    motto: >
      D'oh!! YAML is too complicated!

  - &motto002
    person:   *person002
    motto: >
      Bart! Listen to your father!

  - &motto003
    person:   *person003
    motto: >
      Hey! YAML is freakin' sweet!
```

Problem

- developer wishes to cross-reference rows-with-anchors from one table and link to them with rows-as-aliases in another table

Solution

- use YAML aliases, which cross-reference pre-defined anchors from another table
- in YAML, reusable "transclusion identifiers" are called anchors and aliases
- in YAML, reusable "transclusion identifiers" consist of alphanumeric tokens preceeded by an ampersand or asterisk

Rationale

- YAML anchors and aliases allow for increased data normalization
- YAML anchors and aliases enforce `DRY` (Don't repeat yourself)
- in this example, a table structure can be designed and preserved which closely coincides with a database
- in this example, data entry and file sizes can be reduced

Pitfalls

- in this specific example, `yaml.load()` will produce nested dictionaries
  - this is referred to as the "nested dictionaries problem"
  - under the person name-value pair, the value for person will be a sub-dictionary
  - this may be undesirable, because it breaks the uniformity of the table structure
- failure to correctly specify aliases will result in missing data
  - (typos will create broken cross-references)
- YAML does not support file transclusion by reference, so all aliases and anchors must exist in the same yaml file
- not all YAML parsers reliably support anchors and aliases

See also

Stackoverflow YAML

Using YAML merge-keys to cross-reference rows from another YAML table

```
---
person_table:
  - &person001
    fname:  homer
    lname:  simpson
    role:   dad
    age:    33

  - &person002
    fname:  marge
    lname:  simpson
    role:   mom
    age:    34

  - &person003
```

```
      fname:  peter
      lname:  griffin
      role:   dad
      age:    34

  motto_table:
    - &motto001
      <<: *person001
      motto: >
        D'oh!! YAML is too complicated!

    - &motto002
      <<: *person002
      motto: >
        Bart! Listen to your father!

    - &motto003
      <<: *person003
      motto: >
        Hey! YAML is freakin' sweet!
```

Problem

- developer wishes to cross-reference rows-with-anchors from one table and link to them with rows-as-aliases in another table
- developer wishes to avoid creating the "nested dictionaries problem"

Solution

- use YAML aliases, with YAML merge keys
- in YAML, reusable "transclusion identifiers" are called anchors and aliases
- in YAML, reusable "transclusion identifiers" consist of alphanumeric tokens preceeded by an ampersand or asterisk

Rationale

- YAML anchors and aliases allow for increased data normalization
- YAML anchors and aliases enforce `DRY` (Don't repeat yourself)
- in this example, a table structure can be designed and preserved which closely coincides with a database
- in this example, data entry and file sizes can be reduced

Pitfalls

- in this specific example, `yaml.load()` will produce nested dictionaries
  - under the person name-value pair, the value for person will be a sub-dictionary
  - this may be undesirable, because it breaks the uniformity of the table structure
- failure to correctly specify aliases will result in missing data
  - (typos will create broken cross-references)

- YAML does not support file transclusion by reference, so all aliases and anchors must exist in the same yaml file
- not all YAML parsers reliably support anchors and aliases

See also

- Stackoverflow YAML
- YAML merge-keys specification

Read Using anchors and aliases for transcluded content online:
https://riptutorial.com/yaml/topic/4169/using-anchors-and-aliases-for-transcluded-content

Credits

S. No	Chapters	Contributors
1	Getting started with yaml	Anthon, Community, flyx, James, nus, Paul Sweatte, Praveen Kumar, Steve Bennett
2	Using anchors and aliases for transcluded content	dreftymac