EBook Gratis

APRENDIZAJE yaml

Free unaffiliated eBook created from
Stack Overflow contributors.

#yaml

Tabla de contenido

Acerca de

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

Capítulo 1: Empezando con yaml

Observaciones

YAML es un acrónimo recursivo de " `Y` AML `A` in't `M` arkup `L` angguage". Es un estándar de serialización de datos legible por humanos para todos los lenguajes de programación.

Versiones

Versión	Fecha de lanzamiento
1.0	2004-01-29
1.1	2005-01-18
1.2	2009-10-01

Examples

Sintaxis básica de Yaml

YAML es un formato basado en texto que permite almacenar datos estructurados en una jerarquía. YAML está diseñado para ser humano y legible por máquina con un mínimo de sobrecarga. La especificación YAML se puede encontrar en yaml.org . También hay una tarjeta de referencia.

Los comentarios comienzan con `#` y continúan hasta la nueva línea, los comentarios deben estar separados de otros tokens por espacios en blanco. Los espacios en blanco no son libres, la sangría debe ser espacios, no pestañas. YAML considerará que las líneas prefijadas con más espacios que la clave principal están contenidas dentro de ella. Además, todas las líneas deben estar prefijadas con la misma cantidad de espacios para pertenecer al mismo mapa.

YAML tiene secuencias y mapeos como tipos de colección, ambos pueden representarse en flujo y estilo de bloque.

Una secuencia de cadenas escalares en YAML se parece a:

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

# or block style

- one
- two
- three
```

Un mapeo consiste en pares clave / valor:

```
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
```

Tipos básicos de YAML

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

Datos secuenciales de YAML

Mismo nivel de lista:

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

Lista anidada:

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

Comentarios

```
# 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)
```

Tenga en cuenta que para que un `#` introduzca un comentario, debe

- ocurre al principio de una línea, o
- estar precedido por espacios en blanco.

`#` siempre debe ir seguido de espacios en blanco. `#` dentro de los escalares citados nunca comienzan comentarios. `#` puede introducir comentarios al final de los escalares de bloque, pero, por lo tanto, debe tener menos sangría que la sangría base del escalar de bloque (que generalmente está determinada por la sangría de su primera línea no vacía).

Asignaciones de estilo de bloque

Con claves implícitas:

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

Con claves implícitas y explícitas:

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

`key` , `another key` y `[1, 2, 3]` son claves de la misma asignación, aunque utilizan diferentes estilos de clave.

Asignaciones anidadas:

```
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
```

División de cadenas de texto en múltiples líneas

```
- 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.
```

Personajes que escapan

YAML soporta tres estilos de notación de escape:

1. Entity Escapes

    a. espacio: "& # x20;"

    segundo. colon: "& # 58;"

    do. ampersand: "& amp;"

2. Unicode Escapes

    a. espacio: "\ u0020"

    segundo. comilla simple: "\ u0027"

    do. comillas dobles: "\ u0022"

3. Escapes citados

    a. comillas dobles en comillas simples: '¿Es "siempre miento" una afirmación verdadera?'

    segundo. doble cita anidada: "Ella dijo," renuncié ""

    do. Una sola cita anidada: 'Se quedó sin habla:' "

Lea Empezando con yaml en línea: https://riptutorial.com/es/yaml/topic/3181/empezando-con-yaml

Capítulo 2: Usando anclas y alias para contenido transcluido

Examples

Creación de una tabla "Array of Dictionaries" con anclajes YAML como identificadores de fila

```
---
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
```

Problema

- el desarrollador desea expresar una estructura de tabla en YAML, donde cada fila es referenciada por un identificador de fila compacto

Solución

- use anclajes YAML, asignando un identificador de ancla a cada fila en la tabla
- en YAML, los "identificadores de transclusión" reutilizables se llaman anclas y alias
- en YAML, los "identificadores de transclusión" reutilizables consisten en tokens alfanuméricos precedidos por un signo y un asterisco

Razón fundamental

- Los anclajes y alias de YAML permiten una mayor normalización de los datos
- Los anclajes y los alias de YAML se aplican en `DRY` (no se repita)
- en este ejemplo, se puede diseñar y conservar una estructura de tabla que coincida estrechamente con una base de datos

Escollos

- Los anclajes YAML deben declararse antes de que puedan ser referenciados por alias.
- Los anclajes YAML deben ser únicos en todo el documento.
- si no se especifican los anclajes únicos, se producirá un error en `yaml.load()`
- No todos los analizadores YAML admiten de forma fiable anclajes y alias.

Ver también

YAML Stackoverflow

Uso de alias YAML para hacer referencias cruzadas de una tabla YAML

```
---
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!
```

Problema

- el desarrollador desea hacer una referencia cruzada de filas con anclajes de una tabla y vincularlos con filas como alias en otra tabla

Solución

- use alias YAML, que hacen referencia cruzada a los anclajes predefinidos de otra tabla
- en YAML, los "identificadores de transclusión" reutilizables se llaman anclas y alias
- en YAML, los "identificadores de transclusión" reutilizables consisten en tokens alfanuméricos precedidos por un signo y un asterisco

Razón fundamental

- Los anclajes y alias de YAML permiten una mayor normalización de los datos
- Los anclajes y los alias de YAML se aplican en `DRY` (no se repita)
- en este ejemplo, se puede diseñar y conservar una estructura de tabla que coincida estrechamente con una base de datos
- en este ejemplo, la entrada de datos y el tamaño de los archivos se pueden reducir

Escollos

- en este ejemplo específico, `yaml.load()` producirá diccionarios anidados
  - Esto se conoce como el "problema de diccionarios anidados"
  - debajo del par nombre-valor de la persona, el valor para la persona será un sub-diccionario
  - esto puede ser indeseable, porque rompe la uniformidad de la estructura de la tabla
- Si no se especifican correctamente los alias, se perderán los datos.
  - (Los errores tipográficos crearán referencias cruzadas rotas)
- YAML no admite la transclusión de archivos por referencia, por lo que todos los alias y anclas deben existir en el mismo archivo yaml
- No todos los analizadores YAML admiten de forma fiable anclajes y alias.

Ver también

YAML Stackoverflow

Usando las teclas de combinación YAML para hacer referencias cruzadas de filas de otra tabla YAML

```
---
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!
```

Problema

- el desarrollador desea hacer una referencia cruzada de filas con anclajes de una tabla y vincularlos con filas como alias en otra tabla
- desarrollador desea evitar crear el "problema de diccionarios anidados"

Solución

- usar alias YAML, con claves de fusión YAML
- en YAML, los "identificadores de transclusión" reutilizables se llaman anclas y alias
- en YAML, los "identificadores de transclusión" reutilizables consisten en tokens alfanuméricos precedidos por un signo y un asterisco

Razón fundamental

- Los anclajes y alias de YAML permiten una mayor normalización de los datos
- Los anclajes y los alias de YAML se aplican en `DRY` (no se repita)
- en este ejemplo, se puede diseñar y conservar una estructura de tabla que coincida estrechamente con una base de datos
- en este ejemplo, la entrada de datos y el tamaño de los archivos se pueden reducir

Escollos

- en este ejemplo específico, `yaml.load()` producirá diccionarios anidados
    - debajo del par nombre-valor de la persona, el valor para la persona será un sub-diccionario

    - esto puede ser indeseable, porque rompe la uniformidad de la estructura de la tabla
- Si no se especifican correctamente los alias, se perderán los datos.
    - (Los errores tipográficos crearán referencias cruzadas rotas)
- YAML no admite la transclusión de archivos por referencia, por lo que todos los alias y anclas deben existir en el mismo archivo yaml
- No todos los analizadores YAML admiten de forma fiable anclajes y alias.

Ver también

- YAML Stackoverflow
- Especificación de las claves de fusión YAML

Lea Usando anclas y alias para contenido transcluido en línea:
https://riptutorial.com/es/yaml/topic/4169/usando-anclas-y-alias-para-contenido-transcluido

Creditos

S. No	Capítulos	Contributors
1	Empezando con yaml	Anthon, Community, flyx, James, nus, Paul Sweatte, Praveen Kumar, Steve Bennett
2	Usando anclas y alias para contenido transcluido	dreftymac