[YAML详解](https://www.cnblogs.com/pandaly/p/10321464.html)

<https://www.cnblogs.com/pandaly/p/10321464.html>

# [YAML详解](https://www.cnblogs.com/pandaly/p/10321464.html)

# 1   YAML简介

YAML，即YAML Ain’t Markup Language的缩写，YAML 是一种简洁的非标记语言。YAML以数据为中心，使用空白，缩进，分行组织数据，从而使得表示更加简洁易读。

YAML的[在线Demo](http://nodeca.github.io/js-yaml/)这个YAML转化JSON网页中进行上手练习

# 2   YAML语法

1. 大小写敏感
2. 使用缩进表示层级关系
3. 禁止使用tab缩进，只能使用空格键
4. 缩进长度没有限制，只要元素对齐就表示这些元素属于一个层级。
5. 使用#表示注释
6. 字符串可以不用引号标注

# 3   YAML 组织结构

YAML 文件可以由一或多个文档组成（也即相对独立的组织结构组成），文档间使用“---”（三个横线）在每文档开始作为分隔符。同时，文档也可以使用“...”（三个点号）作为结束符（可选）。如果只是单个文档，分隔符“---”可省略。

# 4   YAML数据结构

YAML 支持的数据结构有三种：

* 对象：键值对的集合，又称为映射（mapping）/ 哈希（hashes） / 字典（dictionary）
* 数组：一组按次序排列的值，又称为序列（sequence） / 列表（list）
* 纯量（scalars）：单个的、不可再分的值

## 4.1    对象

### 4.1.1        使用 “:”（冒号） + 空格表示单个键值对

对象的一组键值对，使用 “:”（冒号） + 空格表示单个键值对，格式为key: value。冒号后面要加一个空格

key: value

可以使用缩进表示层级关系；

key:

    child-key: value

child-key2: value2

key: {child-key: value, child-key2: value2}

### 4.1.2        使用"{}"表示一个键值表

items: {no: 1234, descript: cpu, price: ￥800.00}

### 4.1.3        "? " 问号+空格表示复杂的键

"? " 问号+空格表示复杂的键。当键是一个列表或键值表时，就需要使用本符号来标记。

1. # 使用一个列表作为键
2. ? [blue, reg, green]: Color
3. # 等价于
4. ? - blue
5. - reg
6. - gree
7. : Color

## 4.2    数组

使用一个短横线加一个空格代表一个数组项：

hobby:

    - Java

    - LOL

当然也可以有这样的写法：

-

    - Java

    - LOL

可以简单理解为：[[Java,LOL]]

数组的理解：

PaymentDetail:  
  - billPaymentNo: "201904171022507929513877465876"  
  - paymentAmount: 1000.00  
  - paymentMethod1: "ODD"  
  - paymentSuccessDate: 2019-04-17 13:00:00

转为json如下：

PaymentDetail:[{billPaymentNo: '201904171022507929513877465876' },{paymentAmount: 1000 },{paymentMethod1: 'ODD' },{paymentSuccessDate: Wed Apr 17 2019 21:00:00 GMT+0800 (China Standard Time)}]

理解为：PaymentDetail值为一个列表，列表中有四个元素，其中每个元素都是一个字典，每个字典中只有一个键值对

PaymentDetail1:  
  - billPaymentNo: "201904171022507929513877465876"  
    paymentAmount: 1000.00  
    paymentMethod1: "ODD"  
    paymentSuccessDate: 2019-04-17 13:00:00

转为json如下：

PaymentDetail1: [{billPaymentNo:'201904171022507929513877465876',paymentAmount: 1000, paymentMethod1: 'ODD',paymentSuccessDate: Wed Apr 17 2019 21:00:00 GMT+0800 (China Standard Time)}]

理解为：PaymentDetail1值为一个列表，列表中只有一个元素，该元素为一个字典，字典由多个键值对构成

## 4.3    复合结构

对象和数组可以结合使用，形成复合结构。

2. languages:

1. - Ruby
2. - Perl
3. - Python

6. websites:

1. YAML: yaml.org
2. Ruby: ruby-lang.org
3. Python: python.org
4. Perl: use.perl.org

转为 JavaScript 如下。

1. { languages: [ 'Ruby', 'Perl', 'Python' ],
2. websites:
3. { YAML: 'yaml.org',
4. Ruby: 'ruby-lang.org',
5. Python: 'python.org',
6. Perl: 'use.perl.org' } }

## 4.4    纯量

纯量是最基本的、不可再分的值。以下数据类型都属于纯量：

l  字符串

l  布尔值

l  整数

l  浮点数

l  Null

l  时间

l  日期

## 4.5    特殊用法

### 4.5.1        !! YAML中使用!!做类型强行转换：

yamlbeans包一般用!(单叹号)做类型转换，snakeyaml包一般用!!(双叹号)做类型转换

string:

    - !!str 54321

    - !!str true

相当于把数字和布尔类型强转为字符串。当然允许转型的类型很多，比如：

--- !!set

- Mark McGwire: 65

- Sammy Sosa: 63

- Sammy Sosa: 63

- Ken Griffy: 58

将数组解析为set，转化的内容就是：[{Ken Griffy=58}, {Mark McGwire=65}, {Sammy Sosa=63}]，重复的Sammy Sosa去掉；

### 4.5.2        字符串默认不使用引号表示。

str: 这是一行字符串

### 4.5.3        字符串之中包含空格或特殊字符，需要放在引号之中

如果字符串之中包含空格或特殊字符，需要放在引号之中。单引号和双引号都可以使用

str: '内容： 字符串'

### 4.5.4        双引号不会对特殊字符转义。

s1: '内容\n字符串'

s2: "内容\n字符串"

### 4.5.5        单引号之中如果还有单引号，必须连续使用两个单引号转义。

str: 'labor''s day'

### 4.5.6        多行字符串可以使用|保留换行符，也可以使用>折叠换行。

this: |

  Foo

  Bar

that: >

  Foo

  Bar

转为 JavaScript 代码如下：

{ this: 'Foo\nBar\n', that: 'Foo Bar\n' }

### 4.5.7        +表示保留文字块末尾的换行，-表示删除字符串末尾的换行。

s1: |

  Foo

s2: |+

  Foo

s3: |-

  Foo

转为 JavaScript 代码如下。

{ s1: 'Foo\n', s2: 'Foo\n\n\n', s3: 'Foo' }

### 4.5.8        布尔值用true和false表示。

NULL用~表示。

### 4.5.9        锚点&和别名\*，可以用来引用，定义数据的复用

第一步：使用 “&” 定义数据锚点（即要复制的数据）

第二步：使用 “\*” 引用上述锚点数据（即数据的复制目的地）

2. defaults: &defaults

1. adapter:  postgres
2. host:     localhost

6. development:

1. database: myapp\_development
2. <<: \*defaults
4. test:
5. database: myapp\_test
6. <<: \*defaults

等同于下面的代码。

2. defaults:

1. adapter:  postgres
2. host:     localhost

6. development:

1. database: myapp\_development
2. adapter:  postgres
3. host:     localhost
5. test:
6. database: myapp\_test
7. adapter:  postgres
8. host:     localhost

&用来建立锚点（defaults），<<表示合并到当前数据，\*用来引用锚点。

下面是另一个例子。

2. - &[showell](https://www.baidu.com/s?wd=showell&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd) Steve

3. - Clark

4. - Brian

5. - Oren

6. - \*showell

转为 JavaScript 代码如下。

1. [ 'Steve', 'Clark', 'Brian', 'Oren', 'Steve' ]