Jekyll说明文档

[1安装Jekyll 1](#_Toc3379)

[1.1下载 1](#_Toc28318)

[1.2生成config.yml文件 2](#_Toc13279)

[1.3安装Jekyll 2](#_Toc17540)

[1.4安装bundle 2](#_Toc23178)

[1.5生成博客页面 2](#_Toc17669)

[2文件结构 2](#_Toc12958)

[2.1\_include 2](#_Toc2728)

[2.2\_data 2](#_Toc26684)

[2.3\_layout 2](#_Toc1494)

[2.4\_sass 3](#_Toc5046)

[2.5\_config.yml 3](#_Toc22786)

[2.6\_post 3](#_Toc5000)

[2.7\_site 3](#_Toc32546)

[3参数配置 3](#_Toc31615)

[3.1\_config.yml总的配置 3](#_Toc22232)

[3.2 frontmatter 4](#_Toc27053)

[4 liquid语法 5](#_Toc16380)

[4.1 Advanced output: Filters 5](#_Toc22613)

[4.2 If/Else 6](#_Toc9833)

[4.3 Case Statement 7](#_Toc31743)

[4.4For loops 8](#_Toc27704)

[4.4.1Liquid for遍历循环 8](#_Toc30538)

[4.4.2当遍历一个键值对集合 8](#_Toc30417)

[4.4.3辅助变量 8](#_Toc14906)

[4.4.4limit offset 9](#_Toc32115)

[4.4.5倒叙循环 9](#_Toc20760)

# 1安装Jekyll

## 1.1下载

地址：<https://github.com/pnpdjie/ONAP.git>

windows64位的安装安装Ruby2.3.3(x64)

及[DevKit-mingw64-64-4.7.2-20130224-1432-sfx.exe](https://dl.bintray.com/oneclick/rubyinstaller/DevKit-mingw64-64-4.7.2-20130224-1432-sfx.exe)

## 1.2生成config.yml文件

在Devkit的安装目录下右键GitBash 执行 ruby dk.rb init 生成config.yml文件

继续执行 ruby dk.rb install 安装

## 1.3安装Jekyll

在Devkit的安装目录下右键GitBash 执行 gem install jekyll

若失败，可尝试执行 gem sources --add <http://rubygems.org/>

## 1.4安装bundle

在Devtlt的安装目录下右键Git Bash 执行 gem install bundle

## 1.5生成博客页面

执行 Jekyll new blog生成 初始化页面

执行 Jekyll serve 启动服务。 访问localhost:4000

# 2文件结构

## 2.1\_include

存放一些组件，可以通过{% include file.xx %}引用。常用于header.html footer.html的引用。

## 2.2\_data

存放一些yml配置文件。

## 2.3\_layout

布局文件。如何引用？

## 2.4\_sass

存放主题样式文件，类似css文件

## 2.5\_config.yml

存储配置数据。全局配置的一些参数

## 2.6\_post

存放写的文章，文件名格式:YYYY-MM-DD-TITLE.md

## 2.7\_site

最终生成的静态页面。这部分不用上传github，可以通过 .gitignore来过滤掉 \_site目录

# 3参数配置

## 3.1\_config.yml总的配置

jekyll支持从位于\_data的yaml,json,csv文件中加载数据，（csv必须包含一个header row）  
通过site.data访问里面的数据  
  
例子：  
比如定义一个文件\_data/members.yml  
- name: Tom Preston-Werner  
  github: mojombo  
- name: Parker Moore  
  github: parkr  
- name: Liu Fengyun  
  github: liufengyun

然后可以通过site.data.members访问该文件（文件名决定了字段名）  
<ul>  
{% for member in site.data.members %}  
  <li>  
    <a href="https://github.com/{{ member.github }}">  
      {{ member.name }}  
    </a>  
  </li>  
{% endfor %}  
</ul>  
  
  
定义组织（包含子文件）  
\_data/orgs/jekyll.yml中：  
username: jekyll  
name: Jekyll  
members:  
  - name: Tom Preston-Werner  
    github: mojombo  
  - name: Parker Moore  
    github: parkr

\_data/orgs/doeorg.yml中：  
username: doeorg  
name: Doe Org  
members:  
  - name: John Doe  
    github: jdoe  
  
使用：  
<ul>  
{% for org\_hash in site.data.orgs %}  
{% assign org = org\_hash[1] %}  
  <li>  
    <a href="https://github.com/{{ org.username }}">  
      {{ org.name }}  
    </a>  
    ({{ org.members | size }} members)  
  </li>  
{% endfor %}  
</ul>

## 3.2 frontmatter

Front Matter

通过这个可以设置一些变量（甚至可以自定义变量），比如title  
---  
layout: post  
title: Blogging Like a Hacker  
---  
设置好变量以后，  
你就可以在当前页面或者你的页面依赖的\_layouts或者\_includes  
里的文件通过Liquid 标记，比如{page.title}访问了。

# 4 liquid语法

## 4.1 Advanced output: Filters

输出标记需要的过滤器。过滤器是简单的方法。第一个参数在过滤器的左侧就是过滤器的输入，即需要过滤的内容。过滤器的返回值将是过滤器运行时过滤后的左侧的参数。当没有更多的过滤器，模板会收到结果字符串。

代码示例：

Hello {{ 'tobi' | upcase }}

Hello tobi has {{ 'tobi' | size }} letters!

Hello {{ '\*tobi\*' | textilize | upcase }}

Hello {{ 'now' | date: "%Y %h" }}

Standard Filters标准过滤器：

date -时间格式化

capitalize-设置输入中的某个单词\*

downcase-将输入的字符串转换为小写\*

upcase-将输入的字符串转换为大写

first-获得传入的数组的第一个元素

last-获得传入的数组的最后一个元素

join-用数组的分隔符连接数组中的元素

sort-数组中的元素排序

map-通过指定的属性过滤数组中的元素

size-返回一个数组或字符串的大小

escape-转义一个字符串

escape\_once-返回HTML的转义版本，而不会影响现有的实体转义

strip\_html-从字符串去除HTML

strip\_newlines -从字符串中去除所有换行符（\ n）的

newline\_to\_br-用HTML标记替换每个换行符（\ n）

replace-替换，例如：{{ 'foofoo' | replace:'foo','bar' }} #=> 'barbar'

replace\_first-替换第一个，例如： '{{barbar' | replace\_first:'bar','foo' }} #=> 'foobar'

remove-删除，例如：{{'foobarfoobar' | remove:'foo' }} #=> 'barbar'

remove\_first-删除第一个，例如：{{ 'barbar' | remove\_first:'bar' }} #=> 'bar'

truncate-截取字符串到第x个字符

truncatewords-截取字符串到第x个词

prepend-前置添加字符串，例如：{{ 'bar' | prepend:'foo' }} #=> 'foobar'

append-后置追加字符串，例如：{{'foo' | append:'bar' }} #=> 'foobar'

minus-减法，例如：{{ 4 | minus:2 }} #=> 2

plus-加法，例如：{{'1' | plus:'1' }} #=> '11', {{ 1 | plus:1 }} #=> 2

times-乘法，例如：{{ 5 | times:4 }} #=> 20

divided\_by-除法，例如：{{ 10 | divided\_by:2 }} #=> 5

split-通过正则表达式切分字符串为数组，例如：{{"a~b" | split:"~" }} #=> ['a','b']

modulo-取模，例如：{{ 3 | modulo:2 }} #=> 1

## 4.2 If/Else

{% if user %}

  Hello {{ user.name }}

{% endif %}

# Same as above

{% if user != null %}

  Hello {{ user.name }}

{% endif %}

{% if user.name == 'tobi' %}

  Hello tobi

{% elsif user.name == 'bob' %}

  Hello bob

{% endif %}

{% if user.name == 'tobi' or user.name == 'bob' %}

  Hello tobi or bob

{% endif %}

{% if user.name == 'bob' and user.age > 45 %}

  Hello old bob

{% endif %}

{% if user.name != 'tobi' %}

  Hello non-tobi

{% endif %}

# Same as above

{% unless user.name == 'tobi' %}

  Hello non-tobi

{% endunless %}

# Check for the size of an array

{% if user.payments == empty %}

   you never paid !

{% endif %}

{% if user.payments.size > 0  %}

   you paid !

{% endif %}

{% if user.age > 18 %}

   Login here

{% else %}

   Sorry, you are too young

{% endif %}

# array = 1,2,3

{% if array contains 2 %}

   array includes 2

{% endif %}

# string = 'hello world'

{% if string contains 'hello' %}

   string includes 'hello'

{% endif %}

## 4.3 **Case Statement**

{% case condition %}

{% when 1 %}

hit 1

{% when 2 or 3 %}

hit 2 or 3

{% else %}

... Else ...

{% endcase %}

{% case template %}

{% when 'label' %}

     // {{ label.title }}

{% when 'product' %}

     // {{ product.vendor | link\_to\_vendor }} / {{ product.title }}

{% else %}

     // {{page\_title}}

{% endcase %}

## 4.4For loops

### 4.4.1Liquid for遍历循环

{% for item in array %}

  {{ item }}

{% endfor %}

### 4.4.2当遍历一个键值对集合

item[0]是key的值，item[1]则是value的值。

{% for item in hash %}

  {{ item[0] }}: {{ item[1] }}

{% endfor %}

### 4.4.3辅助变量

在每次for循环中，下面的辅助变量可用于额外的需求：

forloop.length      # => 整个for循环的长度

forloop.index       # => 当前迭代的索引

forloop.index0      # => 当前迭代的索引(从0开始)

forloop.rindex      # => 剩余的迭代次数

forloop.rindex0     # => 剩余的迭代次数(从0开始)

forloop.first       # => 是否是第一次迭代?

forloop.last        # => 是否是最后一次迭代?

### 4.4.4limit offset

Limit限制循环次数 offset:int可以让你从第n项开始遍历。# array = [1,2,3,4,5,6]

{% for item in array limit:2 offset:2 %}

{{ item }}

{% endfor %}

# results in 3,4

### 4.4.5倒叙循环

{% for item in collection reversed %}

    {{item}}

{% endfor %}

## 4.5Variable Assinmengt

将数据存储在自定义的变量，输出或在其他标记中随意使用。用assign标签创建。

{% assign name = 'freestyle' %}

{% for t in collections.tags %}

{% if t == name %}

Freestyle!

{% endif %}

{% endfor %}

{% assign freestyle = false %}

{% for t in collections.tags %}

{% if t == 'freestyle' %}

{% assign freestyle = true %}

{% endif %}

{% endfor %}

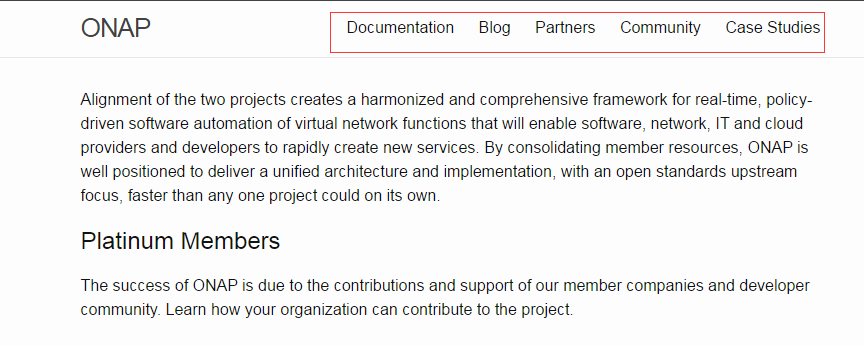
{% if freestyle %}

Freestyle!

{% endif %}

# 5 ONAP demo修改

## 5.1 右上导航



修改位置 \_includes/header.html

