Nginx HTTP Variable

# 概述

# 变量数据结构

## http_variable.pngngx\_http\_variable\_t

表示一个变量，但不包含变量的值。

* name：变量名。
* set\_handler：设置变量值的函数。
* get\_handler：获取变量值的函数。
* data：设置（获取）变量值的函数的参数。
* index：在变量数组的index。
* flags：
  + changeable：变量值不是一个常量，可以改变。
  + nocacheable：不可缓存，每次都要通过get\_handler()获取变量值。
  + indexed：需加入（或已加入）变量数组。
  + nohash：TODO

## ngx\_http\_variable\_value\_t

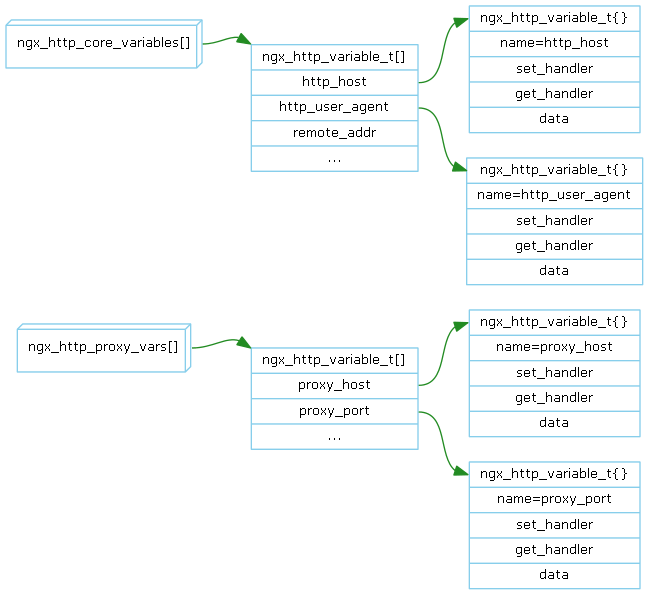
表示一个变量的值，它可以通过变量get\_handler()获取，或者通过set\_handler()设入变量。

它实际上是一个ngx\_str\_t{len, data}，只是多了4个属性。

* valid：变量值有效。
* no\_cacheable：不可缓存。
* not\_found：TODO
* escape：TODO

# 变量申明

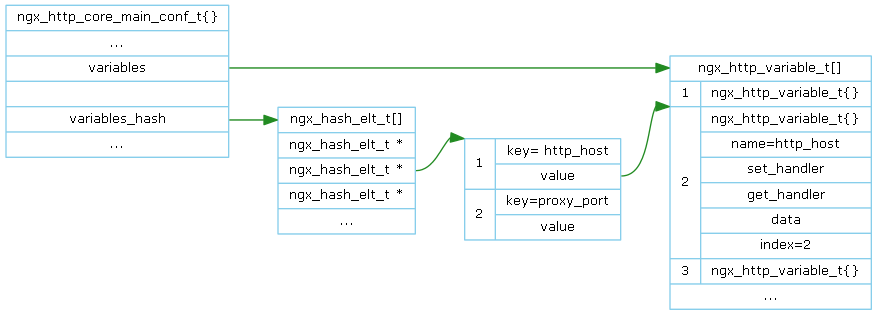
## 内建变量

* ngx\_http\_core\_variables：http\_core模块所定义的变量。
* ngx\_http\_proxy\_vars：http\_proxy模块所定义的变更。

## 外部变量

Rewrite模块的set配置指令。

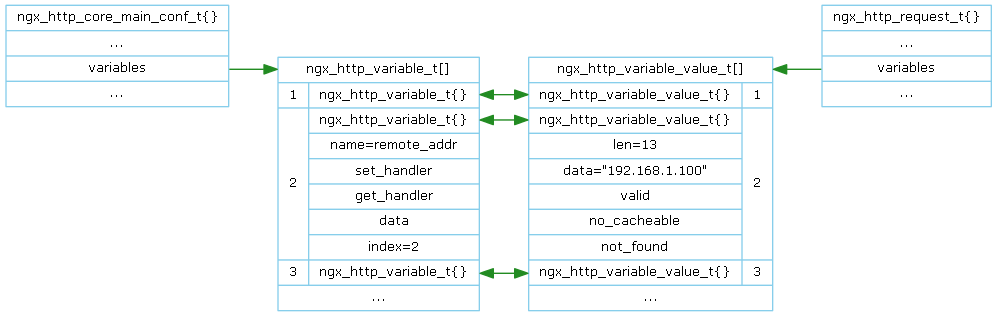
## 变量组织

* variables\_hash：为了快速由变量名查找到ngx\_http\_variable\_t，把已申明的变量建立HASH表。
* variables：将已申明变量加入该数组。

# 变量访问

变量是和Request相关联的。Request是变量的宿主。比如说$remote\_addr变量是某个Request的$remote\_addr。

申明的变量由ngx\_http\_core\_main\_conf\_t的variables\_hash和variables管理。而变量的实例则由ngx\_request\_t的variables管理。



ngx\_http\_variable\_value\_t数组和ngx\_http\_variable\_t数组是一对并列数组，创建ngx\_http\_request\_t时同时也被创建。

ngx\_http\_variable\_value\_t数组的第N个是ngx\_http\_variable\_t数组第N个变量的值。

ngx\_http\_variable\_value\_t \* ngx\_http\_get\_variable(ngx\_http\_request\_t \*r, name, key)函数是获取某个变量的变量值函数。

下面以读变量$remote\_addr值为例，作分析：

1. 根据变量名“remote\_addr”在variables\_hash哈希表中找到ngx\_http\_variable\_t。
2. 根据ngx\_http\_variable\_t的index，得到ngx\_http\_variable\_value\_t。
3. 此时ngx\_http\_variable\_value\_t的valid为0，即无效数据，调用ngx\_http\_variable\_t的get\_handler()
4. ngx\_http\_variable\_remote\_addr()从ngx\_request\_t取得ngx\_connection\_t对象从而取得remote address。
5. 将remote address写入ngx\_http\_variable\_value\_t并设valid为1。
6. 返回ngx\_http\_variable\_value\_t。

此后如果再读该变量的值，因为ngx\_http\_variable\_value\_t的valid为1，即有效，则无需get\_handler()，而直接返回。

# Prototype

暂无

# Latest revision

https://github.com/lingjf/nginx\_analyse/blob/master/doc/