# 模块开发高级篇(30%)

## 变量(80%)

### 综述

在Nginx中同一个请求需要在模块之间数据的传递或者说在配置文件里面使用模块动态的数据一般来说都是使用变量，比如在HTTP模块中导出了host/remote\_addr等变量，这样我们就可以在配置文件中以及在其他的模块使用这个变量。在Nginx中，有两种定义变量的方式，一种是在配置文件中,使用set指令，一种就是上面我们提到的在模块中定义变量，然后导出.

在Nginx中所有的变量都是与HTTP相关的(也就是说赋值都是在请求阶段)，并且基本上是同时保存在两个数据结构中，一个就是hash表(可选)，另一个是数组. 比如一些特殊的变量，比如arg\_xxx/cookie\_xxx等，这些变量的名字是不确定的(因此不能内置)，而且他们还是只读的(不能交由用户修改)，如果每个都要放到hash表中的话(不知道用户会取多少个),会很占空间的，因此这些变量就没有hash,只有索引.这里要注意，用户不能定义这样的变量，这样的变量只存在于Nginx内部.

对应的变量结构体是这样子(每一个变量都是一个ngx\_http\_variable\_s结构体)的：

struct ngx\_http\_variable\_s {  
 ngx\_str\_t name; /\* must be first to build the hash \*/  
 ngx\_http\_set\_variable\_pt set\_handler;  
 ngx\_http\_get\_variable\_pt get\_handler;  
 uintptr\_t data;  
 ngx\_uint\_t flags;  
 ngx\_uint\_t index;  
};

其中name表示对应的变量名字，set/get\_handler表示对应的设置以及读取回调，而data是传递给回调的参数，flags表示变量的属性，index提供了一个索引(数组的脚标)，从而可以迅速定位到对应的变量。set/get\_handler只有在真正读取设置变量的时候才会被调用.

这里要注意flag属性,flag属性就是由下面的几个属性组合而成:

#define NGX\_HTTP\_VAR\_CHANGEABLE 1  
#define NGX\_HTTP\_VAR\_NOCACHEABLE 2  
#define NGX\_HTTP\_VAR\_INDEXED 4  
#define NGX\_HTTP\_VAR\_NOHASH 8

1. NGX\_HTTP\_VAR\_CHANGEABLE表示这个变量是可变的.Nginx有很多内置变量是不可变的，比如arg\_xxx这类变量，如果你使用set指令来修改，那么Nginx就会报错.
2. NGX\_HTTP\_VAR\_NOCACHEABLE表示这个变量每次都要去取值，而不是直接返回上次cache的值(配合对应的接口).
3. NGX\_HTTP\_VAR\_INDEXED表示这个变量是用索引读取的.
4. NGX\_HTTP\_VAR\_NOHASH表示这个变量不需要被hash.

而变量在Nginx中的初始化流程是这样的:

1. 首先当解析HTTP之前会调用ngx\_http\_variables\_add\_core\_vars(pre\_config)来将HTTP core模块导出的变量(http\_host/remote\_addr...)添加进全局的hash key链中.
2. 解析完HTTP模块之后，会调用ngx\_http\_variables\_init\_vars来初始化所有的变量(不仅包括HTTP core模块的变量，也包括其他的HTTP模块导出的变量,以及配置文件中使用set命令设置的变量),这里的初始化包括初始化hash表,以及初始化数组索引.
3. 当每次请求到来时会给每个请求创建一个变量数组(数组的个数就是上面第二步所保存的变量个数)。然后只有取变量值的时候，才会将变量保存在对应的变量数组位置。

### 创建变量

在Nginx中，创建变量有两种方式，分别是在配置文件中使用set指令，和在模块中调用对应的接口，在配置文件中创建变量比较简单，因此我们主要来看如何在模块中创建自己的变量。

在Nginx中提供了下面的接口，可以供模块调用来创建变量。

ngx\_http\_variable\_t \*ngx\_http\_add\_variable(ngx\_conf\_t \*cf, ngx\_str\_t \*name, ngx\_uint\_t flags);

这个函数所做的工作就是将变量 \"name\"添加进全局的hash key表中,然后初始化一些域，不过这里要注意，对应的变量的get/set回调，需要当这个函数返回之后，显式的设置,比如在split\_clients模块中的例子:

var = ngx\_http\_add\_variable(cf, &name, NGX\_HTTP\_VAR\_CHANGEABLE);  
if (var == NULL) {  
 return NGX\_CONF\_ERROR;  
}  
//设置回调  
var->get\_handler = ngx\_http\_split\_clients\_variable;  
var->data = (uintptr\_t) ctx;

而对应的回调函数原型是这样的:

typedef void (\*ngx\_http\_set\_variable\_pt) (ngx\_http\_request\_t \*r,  
 ngx\_http\_variable\_value\_t \*v, uintptr\_t data);  
typedef ngx\_int\_t (\*ngx\_http\_get\_variable\_pt) (ngx\_http\_request\_t \*r,  
 ngx\_http\_variable\_value\_t \*v, uintptr\_t data);

回调函数比较简单，第一个参数是当前请求，第二个是需要设置或者获取的变量值，第三个是初始化时的回调指针，这里我们着重来看一下ngx\_http\_variable\_value\_t,下面就是这个结构体的原型:

typedef struct {  
 unsigned len:28;  
  
 unsigned valid:1;  
 unsigned no\_cacheable:1;  
 unsigned not\_found:1;  
 unsigned escape:1;  
 u\_char \*data;  
} ngx\_variable\_value\_t;

这里主要是data域，当我们在get\_handle中设置变量值的时候，只需要将对应的值放入到data中就可以了，这里data需要在get\_handle中分配内存,比如下面的例子(ngx\_http\_fastcgi\_script\_name\_variable),就是fastcgi\_script\_name变量的get\_handler代码片段:

v->len = f->script\_name.len + flcf->index.len;  
  
v->data = ngx\_pnalloc(r->pool, v->len);  
if (v->data == NULL) {  
 return NGX\_ERROR;  
}  
  
p = ngx\_copy(v->data, f->script\_name.data, f->script\_name.len);  
ngx\_memcpy(p, flcf->index.data, flcf->index.len);

### 使用变量

Nginx的内部变量指的就是Nginx的官方模块中所导出的变量，在Nginx中，大部分常用的变量都是CORE HTTP模块导出的。而在Nginx中，不仅可以在模块代码中使用变量，而且还可以在配置文件中使用。

假设我们需要在配置文件中使用http模块的host变量，那么只需要这样在变量名前加一个$符号就可以了($host).而如果需要在模块中使用host变量，那么就比较麻烦，Nginx提供了下面几个接口来取得变量:

ngx\_http\_variable\_value\_t \*ngx\_http\_get\_indexed\_variable(ngx\_http\_request\_t \*r,  
 ngx\_uint\_t index);  
ngx\_http\_variable\_value\_t \*ngx\_http\_get\_flushed\_variable(ngx\_http\_request\_t \*r,  
 ngx\_uint\_t index);  
ngx\_http\_variable\_value\_t \*ngx\_http\_get\_variable(ngx\_http\_request\_t \*r,  
 ngx\_str\_t \*name, ngx\_uint\_t key);

他们的区别是这样子的，ngx\_http\_get\_indexed\_variable和ngx\_http\_get\_flushed\_variable都是用来取得有索引的变量，不过他们的区别是后一个会处理 NGX\_HTTP\_VAR\_NOCACHEABLE这个标记，也就是说如果你想要cache你的变量值，那么你的变量属性就不能设置NGX\_HTTP\_VAR\_NOCACHEABLE,并且通过ngx\_http\_get\_flushed\_variable来获取变量值.而ngx\_http\_get\_variable和上面的区别就是它能够得到没有索引的变量值.

通过上面我们知道可以通过索引来得到变量值，可是这个索引该如何取得呢，Nginx也提供了对应的接口：

ngx\_int\_t ngx\_http\_get\_variable\_index(ngx\_conf\_t \*cf, ngx\_str\_t \*name);

通过这个接口，就可以取得对应变量名的索引值。

接下来来看对应的例子，比如在http\_log模块中，如果在log\_format中配置了对应的变量，那么它会调用ngx\_http\_get\_variable\_index来保存索引:

static ngx\_int\_t  
ngx\_http\_log\_variable\_compile(ngx\_conf\_t \*cf, ngx\_http\_log\_op\_t \*op,  
 ngx\_str\_t \*value)  
{  
 ngx\_int\_t index;  
 //得到变量的索引  
 index = ngx\_http\_get\_variable\_index(cf, value);  
 if (index == NGX\_ERROR) {  
 return NGX\_ERROR;  
 }  
  
 op->len = 0;  
 op->getlen = ngx\_http\_log\_variable\_getlen;  
 op->run = ngx\_http\_log\_variable;  
 //保存索引值  
 op->data = index;  
  
 return NGX\_OK;  
 }

然后http\_log模块会使用ngx\_http\_get\_indexed\_variable来得到对应的变量值,这里要注意，就是使用这个接口的时候，判断返回值，不仅要判断是否为空，也需要判断value->not\_found,这是因为只有第一次调用才会返回空，后续返回就不是空，因此需要判断value->not\_found:

static u\_char \*  
ngx\_http\_log\_variable(ngx\_http\_request\_t \*r, u\_char \*buf, ngx\_http\_log\_op\_t \*op)  
{  
 ngx\_http\_variable\_value\_t \*value;  
 //获取变量值  
 value = ngx\_http\_get\_indexed\_variable(r, op->data);  
  
 if (value == NULL || value->not\_found) {  
 \*buf = '-';  
 return buf + 1;  
 }  
  
 if (value->escape == 0) {  
 return ngx\_cpymem(buf, value->data, value->len);  
  
 } else {  
 return (u\_char \*) ngx\_http\_log\_escape(buf, value->data, value->len);  
 }  
 }

## upstream

### 使用subrequest访问upstream

### 超越upstream

#### event机制

#### 例讲（主动健康检查模块）

## 使用lua模块