本章讲解 ngx\_http\_request\_t 的成员

**结构**

struct ngx\_http\_request\_s {

    // 请求环境数据（相当于session）

    void                            \*\*ctx;

    // 域的配置

    void                            \*\*main\_conf;

    void                            \*\*srv\_conf;

    void                            \*\*loc\_conf;

    // 内存池

    ngx\_pool\_t                       \*pool;

    // 保存请求头的缓存区

ngx\_buf\_t                        \*header\_in;

    // 请求头数据结构

    ngx\_http\_headers\_in\_t             headers\_in;

    // 响应头数据结构

    ngx\_http\_headers\_out\_t            headers\_out;

    // 请求体数据结构

ngx\_http\_request\_body\_t          \*request\_body;

    // 请求方法对应的 nginx 宏定义

    ngx\_uint\_t                        method;

    // 请求的版本号对应的 nginx 宏定义

ngx\_uint\_t                        http\_version;

    // 请求行

    ngx\_str\_t                         request\_line;

    // 请求的uri (以移除?后面的字符串)

    ngx\_str\_t                         uri;

    // 请求的参数

    ngx\_str\_t                         args;

    // uri 的扩展名

    ngx\_str\_t                         exten;

    // 请求的原始uri (包括?后面的字符串)

    ngx\_str\_t                         unparsed\_uri;

    // 请求方法名 （get|post|...）

    ngx\_str\_t                         method\_name;

    // 请求的版本号

ngx\_str\_t                         http\_protocol;

    // 是否只有头

    unsigned                          header\_only:1;

    // 是否已经发送了响应头

    unsigned                          header\_sent:1;

    ...

};

**请求行**

请求行如：GET /index.html HTTP/1.1

**完整字符串保存**

request\_line字段保存请求行的字符串

**请求方法**

method字段保存请求方法的nginx宏定义

method\_name字段保存请求方法的字符串名

**协议版本号**

http\_version字段保存版本号对应的nginx宏定义

http\_protocol字段保存版本号的字符串

**Url**

    // 请求的uri (以移除?后面的字符串)

    ngx\_str\_t                         uri;

    // 请求的参数

    ngx\_str\_t                         args;

    // uri 的扩展名

    ngx\_str\_t                         exten;

    // 请求的原始uri (包括?后面的字符串)

    ngx\_str\_t                         unparsed\_uri;

资源重定向：

Nginx提供了ngx\_http\_internal\_redirect函数供url重定向

ngx\_int\_t ngx\_http\_internal\_redirect(

    ngx\_http\_request\_t \*r,  // 请求结构

    ngx\_str\_t \*uri,         // 重定向 uri

    ngx\_str\_t \*args);       // 参数

**请求头**

headers\_in字段保存nginx为我们解析好的请求头

typedef struct {

// 保存所有的头元素，元素类型为ngx\_table\_elt\_t

    ngx\_list\_t                        headers;

    // host

ngx\_table\_elt\_t                  \*host;

    ngx\_table\_elt\_t                  \*referer;

    // content-length

    ngx\_table\_elt\_t                  \*content\_length;

    // content-type

    ngx\_table\_elt\_t                  \*content\_type;

    // keep-alive

ngx\_table\_elt\_t                  \*keep\_alive;

    // cookies 字段

ngx\_array\_t                       cookies;

    // content-length 字段数字形式

    off\_t                             content\_length\_n;

    // keep-alive 字段数字形式

    time\_t                            keep\_alive\_n;

    ...

} ngx\_http\_headers\_in\_t;

ngx\_table\_elt\_t是一个键值对的链表元素

**请求体**

request\_body保存请求体结构

**结构**

该结构我们只关心一个成员

typedef struct {

    // 存放请求体的内存区

    ngx\_chain\_t                      \*bufs;

    ...

} ngx\_http\_request\_body\_t;

**操作**

// 读取完成回调函数

typedef void (\*ngx\_http\_client\_body\_handler\_pt)(ngx\_http\_request\_t \*r);

// 异步读取请求体数据

ngx\_int\_t ngx\_http\_read\_client\_request\_body(

    // 请求结构

    ngx\_http\_request\_t \*r,

    // 读取完成回调函数

    ngx\_http\_client\_body\_handler\_pt post\_handler);

**响应头**

headers\_out字段保存响应头数据

**结构**

typedef struct {

    ngx\_list\_t                        headers;

    // 状态码

ngx\_uint\_t                        status;

    size\_t                            content\_type\_len;

    // 响应类型

    ngx\_str\_t                         content\_type;

    // 编码

ngx\_str\_t                         charset;

    // 响应数据长度

    off\_t                             content\_length\_n;

    time\_t                            date\_time;

    // 最后修改时间

    time\_t                            last\_modified\_time;

} ngx\_http\_headers\_out\_t;

**操作**

// 将响应头写入响应中

ngx\_int\_t ngx\_http\_send\_header(ngx\_http\_request\_t \*r);

**响应体**

请求结构并没有提供响应体的成员，但提供了一个函数供我们向响应中写入响应体

// 将in追加到响应中

// in 为响应体数据

ngx\_int\_t ngx\_http\_output\_filter(ngx\_http\_request\_t \*r, ngx\_chain\_t \*in);

// 用于控制响应体发送的函数

ngx\_int\_t ngx\_http\_send\_special(ngx\_http\_request\_t \*r, ngx\_uint\_t flags)

// flags 取值如下：

// 响应体数据写入结束

#define NGX\_HTTP\_LAST   1

// 刷新缓存区，强制发送

#define NGX\_HTTP\_FLUSH  2