OpenResty+LuaJIT 高并发

web 服务实践教程知识点

第三章：Ngxin & OpenResty 基础

第一节：location 匹配和反向代理

语法规则

location [=|~|~*|^~] /uri/ { … }

符号	含义
=	开头表示精确匹配
^~	表示普通字符匹配，如果该选项匹配，只匹配该选项，不匹配别的选项，一般用来匹配目录，nginx 不对 url 做编码
~	开头表示区分大小写的正则匹配
~*	开头表示不区分大小写的正则匹配
!~ 和!~*	分别为区分大小写不匹配及不区分大小写不匹配 的正则
/	通用匹配，任何请求都会匹配到。

多个 location 配置的情况下匹配顺序为（参考资料而来，还未实际验证，试试就知道了，不必拘泥，仅供参考）：

- 首先匹配 `=`
- 其次匹配 `^~`
- 其次是按文件中顺序的正则匹配
- 最后是交给 `/` 通用匹配
- 当有匹配成功时候，停止匹配，按当前匹配规则处理请求

第二节：OpenResty 中的内置变量和自定义变量

参看下表：

名称	说明
$arg_name	请求中的 name 参数
$args	请求中的参数
$binary_remote_addr	远程地址的二进制表示
$body_bytes_sent	已发送的消息体字节数
$content_length	HTTP 请求信息里的"Content-Length"
$content_type	请求信息里的"Content-Type"
$document_root	针对当前请求的根路径设置值
$document_uri	与$uri 相同；比如 /test2/test.php
$host	请求信息中的"Host"，如果请求中没有 Host 行，则等于设置的服务器名
$hostname	机器名使用 gethostname 系统调用的值
$http_cookie	cookie 信息
$http_referer	引用地址
$http_user_agent	客户端代理信息
$http_via	最后一个访问服务器的 Ip 地址。
$http_x_forwarded_for	相当于网络访问路径

名称	说明
$is_args	如果请求行带有参数，返回 “?” ，否则返回空字符串
$limit_rate	对连接速率的限制
$nginx_version	当前运行的 nginx 版本号
$pid	worker 进程的 PID
$query_string	与$args 相同
$realpath_root	按 root 指令或 alias 指令算出的当前请求的绝对路径。其中的符号链接都会解析成真是文件路径
$remote_addr	客户端 IP 地址
$remote_port	客户端端口号
$remote_user	客户端用户名，认证用
$request	用户请求
$request_body	这个变量（0.7.58+）包含请求的主要信息。在使用 proxy_pass 或 fastcgi_pass 指令的 location 中比较有意义
$request_body_file	客户端请求主体信息的临时文件名
$request_completion	如果请求成功，设为"OK"；如果请求未完成或者不是一系列请求中最后一部分则设为空
$request_filename	当前请求的文件路径名，比如 /opt/nginx/www/test.php

名称	说明
$request_method	请求的方法，比如"GET"、"POST"等
$request_uri	请求的 URI，带参数；比如 http://localhost:88/test1/
$scheme	所用的协议，比如 http 或者是 https
$server_addr	服务器地址，如果没有用 listen 指明服务器地址，使用这个变量将发起一次系统调用以取得地址(造成资源浪费)
$server_name	请求到达的服务器名
$server_port	请求到达的服务器端口号
$server_protocol	请求的协议版本，"HTTP/1.0"或"HTTP/1.1"
$uri	请求的 URI，可能和最初的值有不同，比如经过重定向之类的

第三节：OpenResty 中调用 memcached 服务

1.第一个简单 demo，获取 memcached 中指定 key 的值

```
location /foo {
    set $memc_cmd “get”;
    set $memc_key $arg_key;
    memc_pass 127.0.0.1:11211;
 }
```

2.使用 RESTFul API 形式操作 memcached

```
location /foo {
    set $memc_key $arg_key;
    memc_pass 127.0.0.1:11211;
 }
```

3.使用 http 请求中的参数自定义操作类型

```
location /bar {
    set $memc_cmd $arg_cmd;
    set $memc_key $arg_key;
    memc_pass 127.0.0.1:11211;
 }
```

4.使用 memcached 集群

```
http {
    upstream backend {
      server 127.0.0.1:11211;
      keepalive 1024;
    }
```

```
    server {
        ...
        location /memc {
            set $memc_cmd get;
            set $memc_key $arg_key;
            memc_pass backend;
        }
    }
}
```

5.常用参数列表：

参数	说明
$memc_key	Memcached 的 key
$memc_flags	Memcached 的 flag，默认为 0
$memc_exptime	Memcached 过期时间，默认为 0，不过期
$memc_value	Memcached 的值
$memc_cmd	操作 Memcached 的命令，如 get，set 等

6.官方文档：https://github.com/openresty/memc-nginx-module#keep-alive-connections-to-memcached-servers