openresty-lua-redis

参考地址：https://github.com/openresty/lua-resty-redis

1。新建nginx配置文件:nginx-openresty-lua-redis.conf

```
指定配置文件：启动命令
[root@localhost nginx]# ./sbin/nginx -p ./ -c conf/nginx-openresty-lua-redis.conf

文件内容：
[root@localhost nginx]# cat conf/nginx-openresty-lua-redis.conf

worker_processes  1;
error_log logs/error.log;


events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;

    sendfile        on;
    keepalive_timeout  65;

    server {
        listen       8082;
        server_name  localhost;

        location / {
                default_type text/html;
                content_by_lua_file /usr/local/openresty/nginx/lua/lua-openresty-redis.lua;
        }
    }

}
文件目录
[root@localhost nginx]# pwd
/usr/local/openresty/nginx
[root@localhost nginx]#
```

2。修改lua文件：

```
[root@localhost lua]# cat lua-openresty-redis.lua
-- 引用resty的redis
local redis = require "resty.redis";
local red = redis:new();
-- 连接redis
local ok,err = red:connect("127.0.0.1",6379);
if not ok then
        ngx.say("faild to connect",err);
        return
end

ok,err = red:set("dKey","dValue");
if not ok then
        ngx.say("failed to set dKey",err);
        return
end
ngx.say("dKey set dValue success")
return

[root@localhost lua]#
```

读取redis的key对应的值

```
[root@localhost lua]# cat lua-openresty-redis.lua
-- 引用resty的redis
local redis = require "resty.redis";
local red = redis:new();
-- 连接redis
local ok,err = red:connect("127.0.0.1",6379);
if not ok then
        ngx.say("faild to connect",err);
        return
end

ok,err = red:set("dKey","dValue");
if not ok then
        ngx.say("failed to set dKey",err);
        return
end
ok,err = red:get("dKey")
if not ok then
        ngx.say("dKey is null")
else
        ngx.say("dKey's value is :"..ok)
end
```

```
return

[root@localhost lua]#
```

分析OpenResty响应信息：

目的：为了修改以后的响应信息。

```
    server {
        listen 8081;
        location / {
                default_type text/html;
                content_by_lua_block {
                        ngx.say("hi block");
                }
        }
    }
```

获取请求参数信息：

修改nginx-param.conf

```
[root@localhost nginx]# ./sbin/nginx -p ./ -c conf/nginx-param.conf
[root@localhost nginx]# cat conf/nginx-param.conf

worker_processes  1;

error_log  logs/error.log;


events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;
    server {
        listen 8081;
        location / {
                default_type text/html;
                content_by_lua_file /usr/local/openresty/nginx/lua/lua-http-param.lua;
        }
    }
}
[root@localhost nginx]#
```

lua-http-param.lua

```
[root@localhost lua]# cat lua-http-param.lua
-- 获取get请求的参数
local arg = ngx.req.get_uri_args();
for k,v in pairs(arg)
do
        ngx.say("key:",k,"   value:",v);
end
[root@localhost lua]#
```

结合redis实践一下：

```
[root@localhost lua]# cat lua-http-param.lua
-- 获取get请求的参数
local redis = require "resty.redis";
local red = redis:new();
red:connect("127.0.0.1",6379);
-- 省去链接错误的判断，前面课程中有



local arg = ngx.req.get_uri_args();
for k,v in pairs(arg)
do
        ngx.say("key:",k,"   value:",v);
        red:set(k,v);
end
[root@localhost lua]#
```

获取请求头参数

获取http请求中header参数。

lua脚本：

```
[root@localhost lua]# cat lua-header-param.lua
-- 获取header参数
local headers = ngx.req.get_headers();
for k,v in pairs(headers)
```

```
do
        ngx.say("[header] key:",k," value:",v);
end
[root@localhost lua]#
```

nginx配置修改

```
[root@localhost conf]# cat nginx-param.conf

worker_processes  1;

error_log  logs/error.log;


events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;
    server {
        listen 8081;
        location / {
                default_type text/html;

                content_by_lua_file /usr/local/openresty/nginx/lua/lua-header-param.lua;

        }
    }



}
[root@localhost conf]#
```

获取post body 键值对 参数

nginx配置文件

```
[root@localhost conf]# cat nginx-param.conf

worker_processes  1;

error_log  logs/error.log;
```

```
events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;
    server {
        listen 8081;
        location / {
                default_type text/html;

                content_by_lua_file /usr/local/openresty/nginx/lua/lua-post-kv-param.lua;



        }
    }


}

[root@localhost conf]#
```

lua代码

```
[root@localhost lua]# cat lua-post-kv-param.lua
-- 获取post body kv参数
-- 重要：读取body
ngx.req.read_body();
local postArgs = ngx.req.get_post_args();
for k,v in pairs(postArgs)
do
        ngx.say("[post] key:",k," value:",v);
end
[root@localhost lua]#
```

获取body体信息

lua脚本：

```
[root@localhost lua]# cat lua-post-body-param.lua
-- 获取body体参数
-- 所有获取body的操作，这个很重要
ngx.req.read_body();
```

```
local body = ngx.req.get_body_data();
ngx.say(body);
[root@localhost lua]#
```

nginx配置文件

```
[root@localhost conf]# cat nginx-param.conf

worker_processes  1;

error_log  logs/error.log;



events {
    worker_connections  1024;
}



http {
    include       mime.types;
    default_type  application/octet-stream;
    server {
        listen 8081;
        location / {
                default_type text/html;

                content_by_lua_file /usr/local/openresty/nginx/lua/lua-post-body-param.lua;



        }
    }



}

[root@localhost conf]#
```