Nginx+Lua入门知识

> OpenResty （也称为 ngx_openresty）是一个全功能的 Web 应用服务器。它打包了标准的 Nginx核心，很多的常用的第三方模块，以及它们的大多数依赖项。
>
> 通过众多进行良好设计的 Nginx 模块，OpenResty 有效地把 Nginx 服务器转变为一个强大的 Web 应用服务器，基于它开发人员可以使用 Lua 编程语言对 Nginx 核心以及现有的各种 Nginx C 模块进行脚本编程，构建出可以处理一万以上并发请求的极端高性能的 Web 应用。
>
> OpenResty 致力于将你的服务器端应用完全运行于 Nginx 服务器中，充分利用 Nginx 的事件模型来进行非阻塞 I/O 通信。不仅仅是和 HTTP 客户端间的网络通信是非阻塞的，与MySQL、PostgreSQL、Memcached、以及 Redis 等众多远方后端之间的网络通信也是非阻塞的。
>
> 因为 OpenResty 软件包的维护者也是其中打包的许多 Nginx 模块的作者，所以 OpenResty 可以确保所包含的所有组件可以可靠地协同工作。

可直接通过官网安装，官网地址：http://openresty.org/

> lua for windows其实是一整套Lua的开发环境.Lua for Windows 为 Windows 系统下提供了 Lua 脚本语言的开发和运行环境。Lua 是一个小巧的脚本语言。作者是巴西人。该语言的设计目的是为了嵌入应用程序中，从而为应用程序提供灵活的扩展和定制功能。
>
> Lua脚本可以很容易的被C/C++代码调用，也可以反过来调用C/C++的函数，这使得Lua在应用程序中可以被广泛应用。不仅仅作为扩展脚本，也可以作为普通的配置文件，代替XML,Ini等文件格式，并且更容易理解和维护。
>
> Lua由标准C编写而成，代码简洁优美，几乎在所有操作系统和平台上都可以编译，运行。 一个完整的Lua解释器不过200k，在目前所有脚本引擎中，Lua的速度是最快的。这一切都决定了Lua是作为嵌入式脚本的最佳选择。

Win下载地址：http://code.google.com/p/luaforwindows/

Hello Lua

nginx通过content_by_lua和content_by_lua_file来嵌入lua脚本。

content_by_lua

修改nginx配置文件nginx.conf

```
location /hellolua {
    content_by_lua '
        ngx.header.content_type = "text/html";
        ngx.say("Hello Lua.");
    ';
}
```

重启nginx访问 http://localhost//hellolua 应该可以看到 Hello Lua.

content_by_lua_file

```
location /demo {
    lua_code_cache off;
    content_by_lua_file lua_script/demo.lua;
}
```

lua_code_cache表示关掉缓存，缓存关掉的情况下修改lua脚本不需要重启nginx。content_by_lua_file指定脚本路径。此处为相对路径，相对于nginx根目录，然后写上下面lua脚本

```
-- filename:demo.lua
ngx.header.content_type = "text/html"
ngx.say("Hello Lua Demo.")
```

重启Nginx，关掉lua_code_cache后nginx会有个alert。

```
nginx: [alert] lua_code_cache is off; this will hurt performance in ./conf/nginx.conf:56
```

访问 http://localhost/demo 则可以看到 Hello Lua Demo.

Nginx常用参数获取

```
ngx.header.content_type = "text/html"
ngx.header.PowerBy = "Lua"
-- 请求头table
for k, v in pairs(ngx.req.get_headers()) do
    ngx.say(k, ": ", v)
end

-- 请求方法 GET、POST等
ngx.say("METHOD:" .. ngx.var.request_method)

-- 获取GET参数
for k, v in pairs(ngx.req.get_uri_args()) do
    ngx.say(k, ":", v)
end


-- 获取POST参数
ngx.req.read_body()
for k, v in pairs(ngx.req.get_post_args()) do
    ngx.say(k, ":", v)
end

-- HTTP版本
ngx.say(ngx.req.http_version())

-- 未解析的请求头字符串
ngx.say(ngx.req.raw_header())

-- 未解析的BODY字符串
ngx.print(ngx.req.get_body_data())

-- ngx.exit(400)
-- ngx.redirect("/", 200)
```

获取MD5示例

下面看个小例子，生成字符串的md5值。

```
ngx.header.content_type = "text/html"
local resty_md5 = require "resty.md5"
local  md5 = resty_md5:new()

local s = "Hello Lua."
md5:update(s)
local str = require "resty.string"
ngx.say(str.to_hex(md5:final()))

ngx.say(ngx.md5(s))
```

-- EOF -- 标签： Lua 发表于： 2016-01-01 16:15

Github地址：https://github.com/pengbotao/itopic.go/blob/master/posts/lua/Nginx+Lua入门知识.md