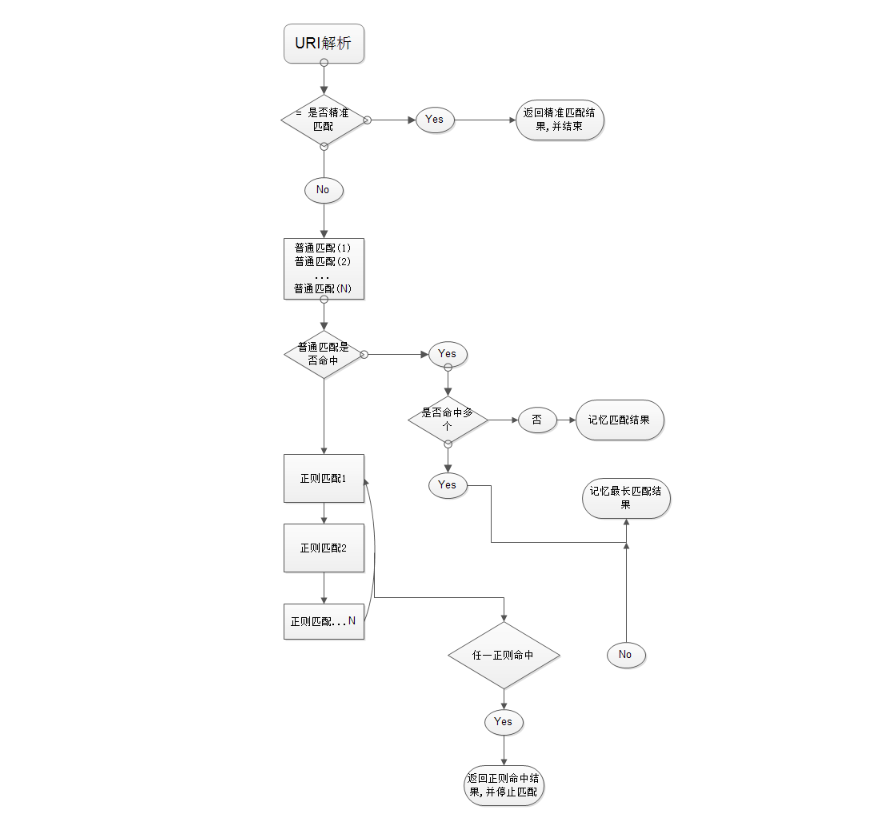
# 七月学习报告

本月主要学习openresty基础知识。对常用的几个点做详细的笔记：

### [location匹配规则](http://nginx.org/en/docs/http/ngx_http_core_module.html#location)

语法: location [ = | ~ | ~\* | ^~ ] uri { ... }

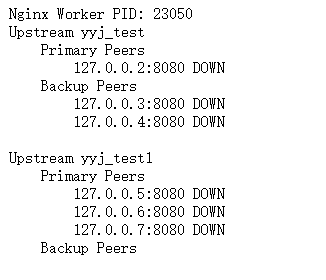
* 1. 优先精准匹配=
  2. 普通匹配，遵循最长匹配原则
  3. 继续正则匹配（如有= or ^~修辞，则不执行正则匹配）
  4. 以上步骤未匹配到，则匹配到 /
  5. 示意图



### [后端服务健康检查](https://github.com/openresty/lua-resty-upstream-healthcheck)

* 1. 配置
     1. 通过init\_worker\_by\_lua\* 当nginx初始化时，开始加载
     2. 配置lua\_shared\_dict 共享内存区域
     3. 导入resty.upstream.healthcheck模块，调用spawn\_checker(optionals) 和 status\_page()接口
  2. spawn\_checker(optionals)
     1. shm 所有工作线程，共享内存
     2. upstream upstream名称（单个）
     3. type 请求协议http
     4. http\_req 请求路径
     5. interval 循环检查时间ms
     6. timeout 网络超时ms
     7. fall down之前连续失败次数
     8. rise up之前连续成功次数
     9. valid\_statuses 合法状态码的数组
     10. concurrency 创建该数目的轻量线程来并发发送健康检测请求的个数
  3. status\_page()

返回格式



### [rewrite规则](http://nginx.org/en/docs/http/ngx_http_rewrite_module.html#rewrite)

* 1. 语法：rewrite regex replacement [flag];
  2. 执行顺序（循环超过10次，则返回500 Internal Server Error错误）
     1. 执行server块的rewrite指令
     2. 执行location匹配
     3. 执行选定的location中的rewrite指令
  3. Flag
     1. last 结束当前匹配，开始新一次
     2. break 结束匹配
     3. redirect 302临时重定向
     4. permanent 301永久重定向

### 内置绑定变量

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

$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使用 Nginx 内置绑定变量206

$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

$request\_method 请求的方法，比如"GET"、"POST"等

$request\_uri 请求的URI，带参数

$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

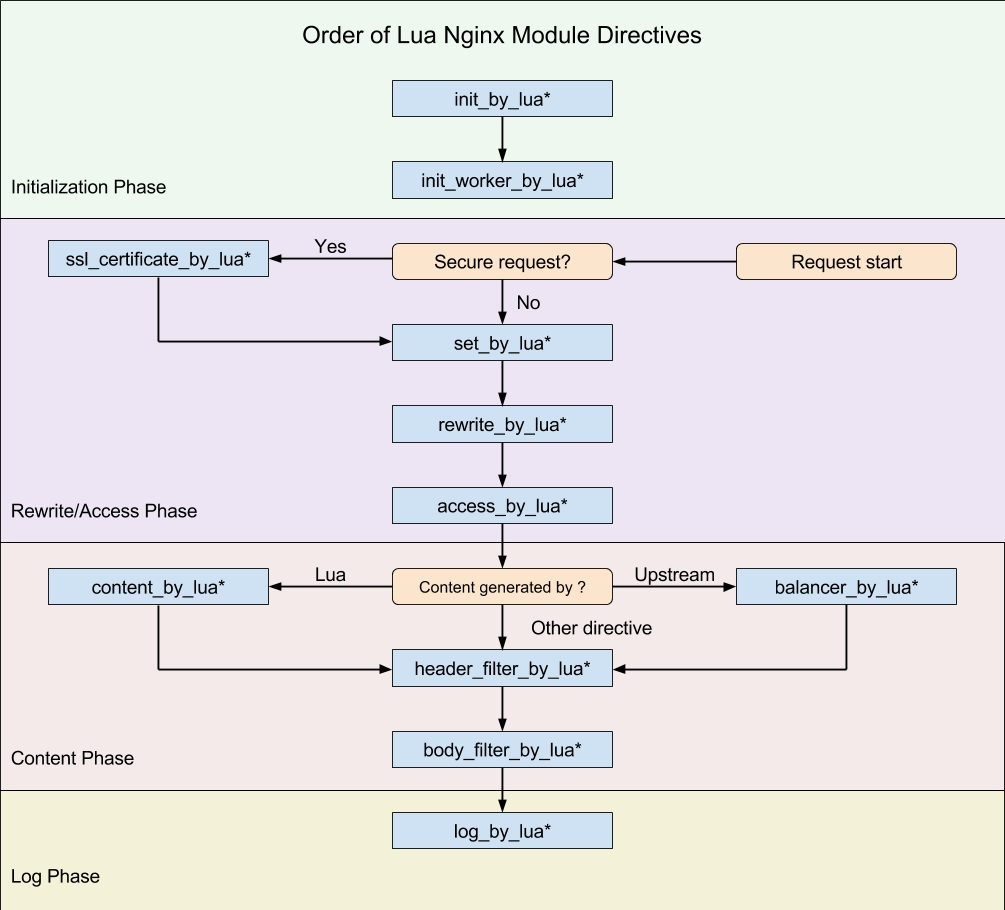
$server\_port 请求到达的服务器端口号

$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

### [openresty执行阶段](https://github.com/openresty/lua-nginx-module#init_by_lua)

详见示意图。



set\_by\_lua\* : 流程分支处理判断变量初始化

rewrite\_by\_lua\* : 转发、重定向、缓存等功能(例如特定请求代理到外网)

access\_by\_lua\* : IP 准入、接口权限等情况集中处理(例如配合 iptable 完成简单防火墙)

content\_by\_lua\* : 内容生成

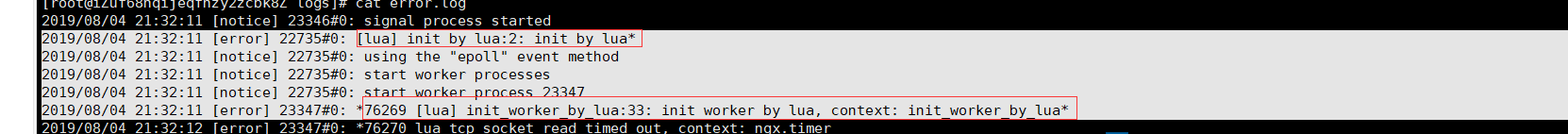
header\_filter\_by\_lua\* : 响应头部过滤处理(例如添加头部信息)

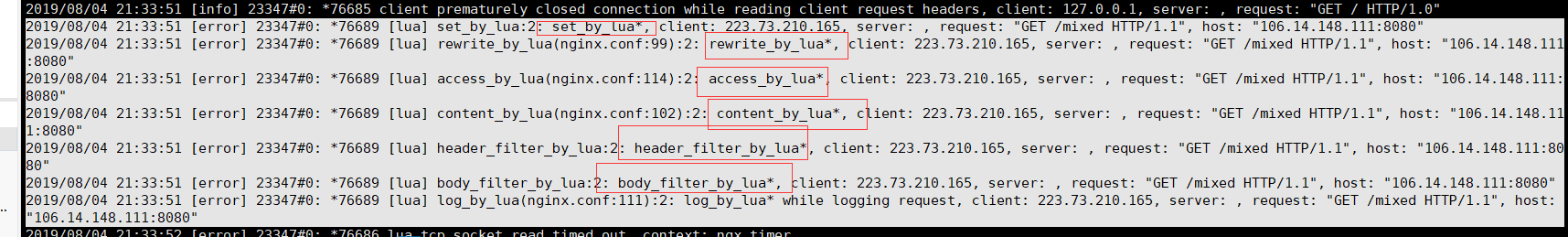
body\_filter\_by\_lua\* : 响应体过滤处理(例如完成应答内容统一成大写)

log\_by\_lua\* : 会话完成后本地异步完成日志记录(日志可以记录在本地，还可以同步到其

他机器)

打乱顺序测试结果如下：





### [nginx执行阶段](https://github.com/nginx/nginx/blob/master/src/http/ngx_http_core_module.h)

NGX\_HTTP\_POST\_READ\_PHASE:

接收完请求头之后的第一个阶段，它位于uri重写之前，实际上很少有模块会注册在该阶段，默认的情况下，该阶段被跳过

NGX\_HTTP\_SERVER\_REWRITE\_PHASE:

server级别的uri重写阶段，也就是该阶段执行处于server块内，location块外的重写指令，在读取请求头的过程中nginx会根据host及端口找到对应的虚拟主机配置

NGX\_HTTP\_FIND\_CONFIG\_PHASE:

寻找location配置阶段，该阶段使用重写之后的uri来查找对应的location，值得注意的是该阶段可能会被执行多次，因为也可能有location级别的重写指令

NGX\_HTTP\_REWRITE\_PHASE:

location级别的uri重写阶段，该阶段执行location基本的重写指令，也可能会被执行多次

NGX\_HTTP\_POST\_REWRITE\_PHASE:

location级别重写的后一阶段，用来检查上阶段是否有uri重写，并根据结果跳转到合适的阶段

NGX\_HTTP\_PREACCESS\_PHASE:

访问权限控制的前一阶段，该阶段在权限控制阶段之前，一般也用于访问控制，比如限制访问频率，链接数等

NGX\_HTTP\_ACCESS\_PHASE:

访问权限控制阶段，比如基于ip黑白名单的权限控制，基于用户名密码的权限控制等

NGX\_HTTP\_POST\_ACCESS\_PHASE:

问权限控制的后一阶段，该阶段根据权限控制阶段的执行结果进行相应处理

NGX\_HTTP\_TRY\_FILES\_PHASE:

try\_files指令的处理阶段，如果没有配置try\_files指令，则该阶段被跳过

NGX\_HTTP\_CONTENT\_PHASE:

内容生成阶段，该阶段产生响应，并发送到客户端

NGX\_HTTP\_LOG\_PHASE:

日志记录阶段，该阶段记录访问日志

### 总结

本月主要熟悉openresty的一些常见配置，以及lua语法。

下月计划学习openresty常见的模块、luajit语法以及Java多线程。