目录

[1 相同severname下的多个虚拟主机优先级 1](#_Toc18217)

[2 location匹配优先级 1](#_Toc2838)

[0 假设 1](#_Toc10450)

[1 图像说明 2](#_Toc19012)

[2 实际例子说明 2](#_Toc4210)

[3 详解案例 3](#_Toc12042)

[3 try\_files使用 4](#_Toc28105)

[1 定义 4](#_Toc22567)

[2 例子 4](#_Toc691)

[3 企业实际用于缓存配置例子 5](#_Toc28108)

[4 nginx下alias和root的区别 5](#_Toc13625)

[5 用什么方案传递真实的ip 6](#_Toc10747)

# 1 相同severname下的多个虚拟主机优先级

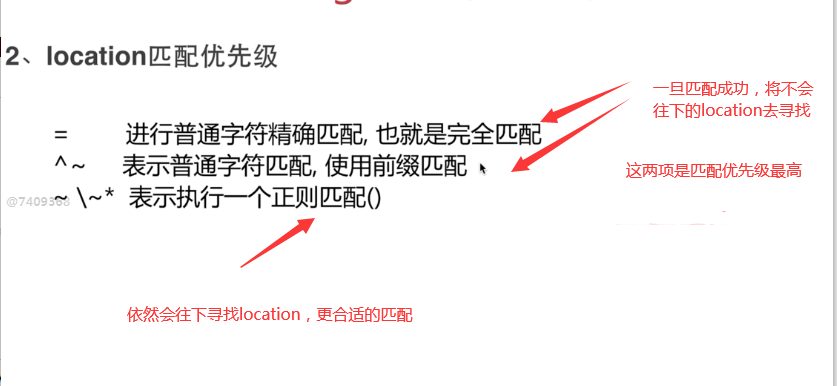


# 2 location匹配优先级

## 0 假设

前端访问jeson.t.imoocc.io/code1/

## 1 图像说明



^~ 前缀字符串匹配，如下方的^~/code,将匹配到/code1,/code2等情况。不会继续向下继续匹配。

~ \~\* 正则匹配，排行第3，找到了这个匹配后，还会继续往下寻找更合适的匹配。这是和以上的匹配不同的地方。

## 2 实际例子说明



## 3 详解案例

location / {

return 401;

}

location = / {

return 402;

}

location = /images/ {

return 501;

}

location /documents/ {

return 403;

}

location ^~ /images/ {

return 404;

}

location ~\* \.(gif|jpg|jpeg)$ {

return 500;

}

**正则匹配结果如下：**

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com](http://www.yunjisuan.com)

402 #location = /

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/](http://www.yunjisuan.com/)

402 #location = /

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/xxxx](http://www.yunjisuan.com/xxxx)

401 #匹配不到,location /

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/documents/](http://www.yunjisuan.com/documents/)

403 #location /documents/

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/images/](http://www.yunjisuan.com/images/)

501 #优先匹配=的情况,location = /images/

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/images/1.jpg](http://www.yunjisuan.com/images/1.jpg)

404 #location ^~ /images/

curl -s -o /dev/null -w "%{http\_code}\n" [www.yunjisuan.com/documents/images/1.jpg](http://www.yunjisuan.com/documents/images/1.jpg)

500 # location ~\* \.(gif|jpg|jpeg)$

**从多个location的配置匹配可以看出匹配的优先顺序**

| **顺序** | **匹配标识的location** | **匹配说明** |
| --- | --- | --- |
| 1 | " location = / { " | 精确匹配 |
| 2 | " location ^~ /images/ { " | 先进行字符串的前缀匹配，如果匹配到就不做正则匹配检查 |
| 3 | " location ~\* \.(gif | jpg | jpeg)$ { " | 正则匹配，\*为不区分大小写 |
| 4 | " location /documents/ { " | 匹配常规字符串，模糊匹配，如果有正则检查，正则优先 |
| 5 | " location / { " | 所有location都不能匹配后的默认匹配原则 |

# 3 try\_files使用

## 1 定义

按顺序检测文件是否存在

## 2 例子

Location /{

try\_files $uri $uri/ index.php

}

首先如果$uri是文件，则返回文件

如果不是，则尝试目录/看是否存在

如果都不是，则交回index.php

## 3 企业实际用于缓存配置例子



# 4 nginx下alias和root的区别

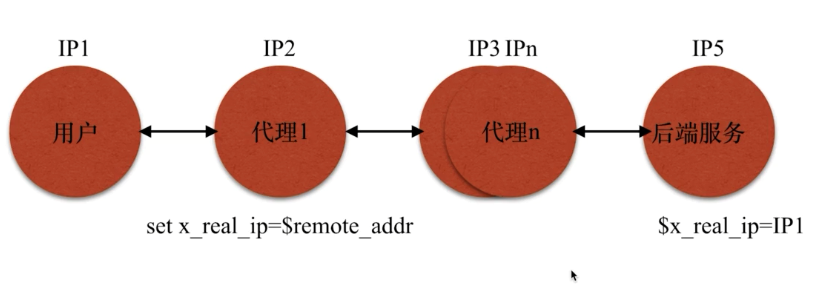
1 root使用



2 alias使用



# 5 用什么方案传递真实的ip



向一级代理设定 X\_real\_ip

经过层层代理透传头信息到后端服务器

后端就能拿到真实的ip