13 找到你的Mr Right：Location 查找原则

更新时间：2020-01-07 09:36:21

> 你若要喜爱你自己的价值，你就得给世界创造价值。——歌德

前言

在前面的文章中，我们做了很多铺垫，比如Nginx配置文件的结构，PCRE正则表达式等等，只有大家弄明白了这些东西，才能深入的学习后面的知识。
我们一直在强调，Nginx是一个优秀的HTTP服务器，那么作为一个服务器，它必须要能够根据用户的不同请求进行不同的处理，location指令就是为了完成这个需求的。

location的用法

location是nginx的一大利器，该指令可以让根据请求的URI分别进行不同的处理，比如如果用户请求的是一个图片，那么需要到pic目录下面寻找，如果请求的是视频，那么需要到video目录下面寻找，这样我们就可以把不同类型的文件分开存储，方便了管理。
我们先来看一下location的语法规则，非常的简单：

```
Syntax: location [ = | ~ | ~* | ^~ ] uri { ... }
        location @name { ... }
Default: —
Context: server, location
```

我把这几种情况进行了分类:

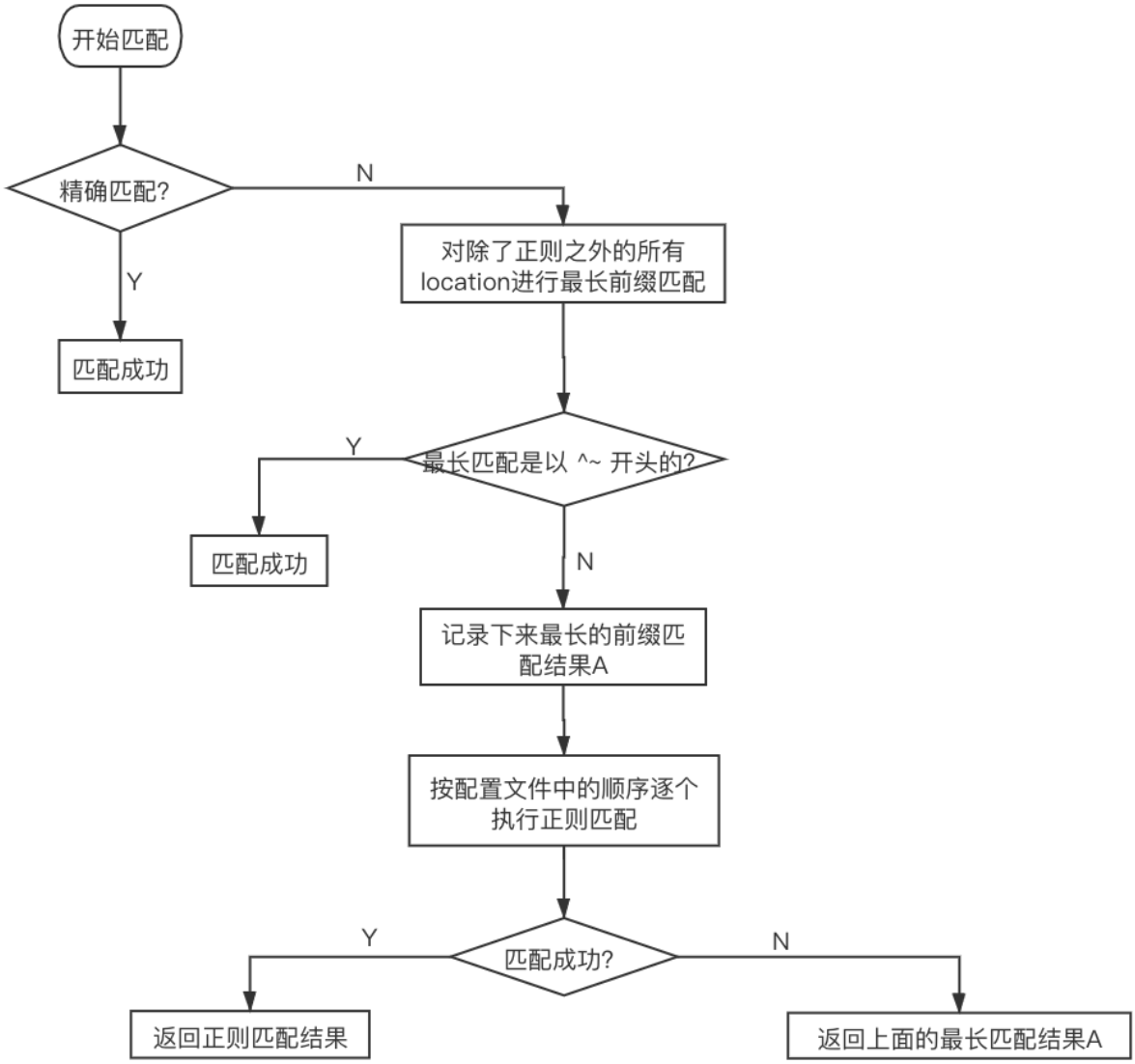
1. 普通匹配(没有任何修饰符)
2. 精确匹配（以"="开头）
3. 最长前缀匹配(以"^~"开头)
4. 正则匹配

    - 区分大小写的正则(以~开头)
    - 忽略大小写的正则(以~*开头)

5. 内部location(以@开头)

匹配顺序

我在工作过程中发现有很多人对Location的匹配顺序搞不清楚，所以这节课我就会以着重给大家分析一下如何根据用户请求的URI进行Location匹配。



上图是我画的一个流程图，从这个流程图中我们可以清晰的看到location的匹配顺序。
从流程图中我们可以总结如下几点:

1. 精确匹配的优先级最高。
2. 如果没有精确匹配，那么就会对配置文件中的所有非正则location进行匹配，找到最长匹配。3. 如果最长匹配是以^~开头，那么就返回该匹配结果。

3. 对正则匹配逐个进行匹配，如果匹配成功，则返回正则 location，如果不成功，则返回第 2 步匹配的最长匹配结果。

> 划重点了：
>
> 1. 第 2 步是要对所有的非正则 location 都要进行匹配，也就是说，非正则匹配与 location 出现的顺序无关
> 2. 第四步是对正则 location 逐个匹配，如果成功就直接返回，所有正则表达式结果与配置文件中出现的顺序有关系。

例子走起来

我们用一个实际的例子来说明整个过程：

```
18          location / {
19                  return 200 "/\n";
20          }
21
22          location ^~/first {
23                  return 200 "^~/first\n";
24          }
25
26          location /first/abc/def {
27                  return 200 "/first/abc/def/\n";
28          }
29
30          location ~ ^/first/abc {
31                  return 200 "~ ^/first/abc\n";
32          }
33
34          location ^~ /first/a {
35                  return 200 "^~/first/a\n";
36          }
37
38          location ^~ /first/abc {
39                  return 200 "^~/first/abc\n";
40          }
41
42          location =/first {
43                  return 200 "=/first\n";
44          }
```

上面是我们的一个配置文件，我们测试一下，看看是不是按照我们图中的顺序进行匹配的。

精确匹配

```
[root@bb462a4ca297 /]# curl -i http://localhost/first
HTTP/1.1 200 OK
Server: nginx/1.16.1
Date: Sun, 05 Jan 2020 23:48:56 GMT
Content-Type: application/octet-stream
Content-Length: 8
Connection: keep-alive

=/first
[root@bb462a4ca297 /]#
```

精确匹配

从结果分析，我们访问的路径是`/first`，虽然配置文件中精确匹配的`location`在配置文件的最后面，但还是匹配到了精确匹配，这也说明精确匹配的优先级是最高的。

最长前缀匹配

```
[root@bb462a4ca297 /]# curl -i http://localhost/first/abc
HTTP/1.1 200 OK
Server: nginx/1.16.1
Date: Mon, 06 Jan 2020 00:07:38 GMT
Content-Type: application/octet-stream
Content-Length: 13
Connection: keep-alive

^~/first/abc
```

从结果中可以看出来，返回的结果是`最长前缀匹配`，并没有进行正则匹配。

正则匹配

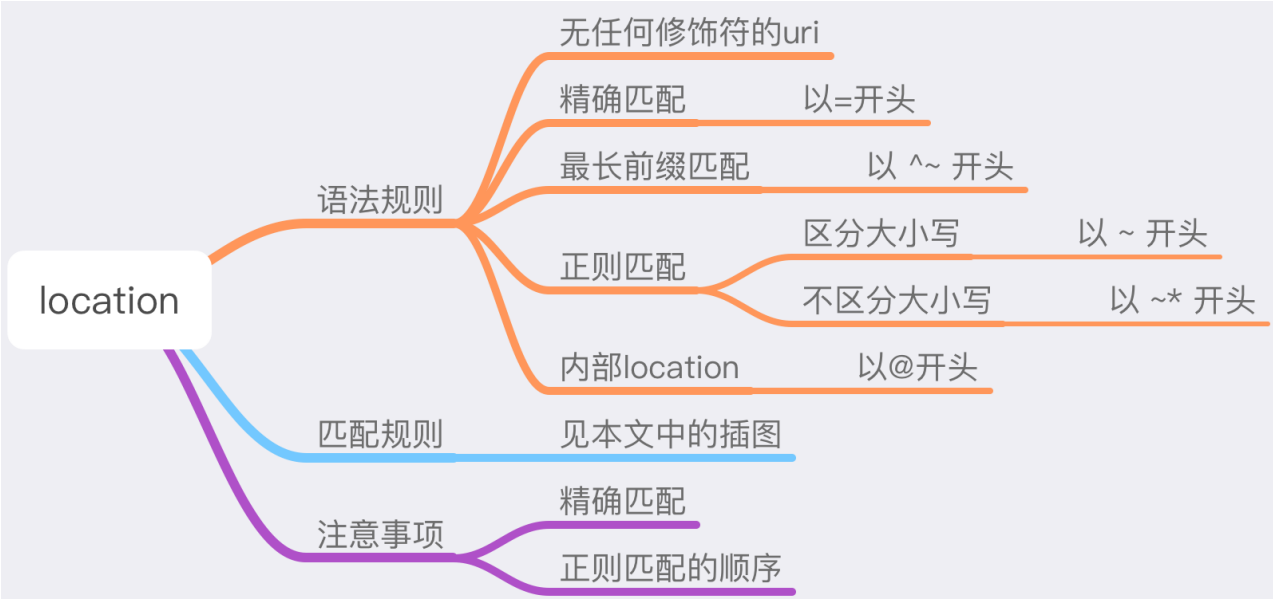
```
[root@bb462a4ca297 /]#
[root@bb462a4ca297 /]# curl -i http://localhost/first/abc/def
HTTP/1.1 200 OK
Server: nginx/1.16.1
Date: Mon, 06 Jan 2020 00:10:49 GMT
Content-Type: application/octet-stream
Content-Length: 12
Connection: keep-alive

~ ^/first/a
[root@bb462a4ca297 /]#
```

我们首先按照流程图可以知道，当前请求的最长前缀匹配应该是`/abc/def`，但是为什么返回了`~ ^/first/a`呢？
因为后来进行了正则匹配，并且`~ ^/first/a`是第一个正则匹配，匹配成功之后直接返回了，所以后面的`~ ^/first/abc`正则没有被执行。

> 这里有两点小建议：
>
> 1. 因为精确匹配的优先级最高，比如根域名这些经常被访问的URI建议使用精确匹配。
> 2. 正则匹配与顺序有关，所以建议越详细的正则应该越靠前。

小结

好了，关于 Location 查找我们就先讲到这里，希望对你有所帮助~



}

12 深入浅出 Rewrite 语法

14 保留证据：能玩出花来的 Nginx 日志