

# 【Nginx】location 匹配規則

## 一般匹配

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

模式	含义
location = /uri	= 表示精确匹配，只有完全匹配上才能生效
location ^~ /uri	^~ 以URL路径行前匹配，并且在正之前。
location ~ pattern	以表示分大小的正匹配
location ~* pattern	以表示不分大小的正匹配
location /uri	不任何修饰符，也表示前匹配，但是在正匹配之后
location /	通用匹配，任何未匹配到其它location的请求都会匹配到，相当于switch中的default

實際使用規則至少有三個匹配定義

```
# 直接匹配网站根，通域名网站首页比繁，使用个会加速处理，官网如是。 # 里是直接发后端用服务器了，也可以是一个首页
# 第一个必
location = / {
    proxy_pass http://tomcat:8080/index
}
# 第二个必是处理文件求，是 nginx 作 http 服务器的强
# 有两种配置模式，目匹配或后匹配，任其一或搭配使用
location ^~ /static/ {
    root /webroot/static;
}
location ~* \.(gif|jpg|jpeg|png|css|js|ico)$ {
    root /webroot/res;
}
# 第三个就是通用，用发动求到后端用服务器
# 非文件求就默是动求，自己根据实际把握
# 竟目前的一些框架的流行，.php、.jsp后的情况很少了 location / {
    proxy_pass http://tomcat:8080/
}
```

依據檔案類型設置過期時間

```
location ~* \.(js|css|jpg|jpeg|gif|png|swf)$ {
    if (-f $request_filename) {
        expires 1h;
        break; }
}
```

禁止訪問某目錄

```
location ~* \.(txt|doc){
    root /data/www/wwwroot/linuxtone/test;
    deny all;
}
```

## 內部命名位置

### 命名位置的用途

- Ⓢ 開頭: 這個符號用來表示這是一個內部使用的命名位置，不會與普通 URL 路徑混淆。
- 命名自由: 你可以給命名位置起任何名稱，只要以 Ⓢ 開頭並且不與其他位置塊或指令名稱衝突。

### 如何使用命名位置

命名位置常用於內部重定向，當某些條件滿足時會跳轉到這些位置。通常用在 `try_files`、`error_page` 等指令中。

## 命名位置的典型應用

- **靜態資源**: 先查找靜態文件，如果找不到則交給動態處理程序處理。
- **錯誤處理**: 如果遇到特定錯誤碼，重定向到特定的錯誤處理位置。
- **代理後備**: 如果特定條件下需要將請求代理到其他服務器。

## 使用 `try_files` 與命名位置

```
server {
    listen 80;
    server_name example.com;

    # 主位置塊
    location / {
        # 嘗試匹配靜態文件，如果找不到則重定向到 @backend
        try_files $uri $uri/ @backend;
    }

    # 定義 @backend 內部重定向位置
    location @backend {
        # 添加自定義標頭
        add_header X-Cache-Status "MISS" always;
        # 設定代理地址
        proxy_pass http://backend_server;
    }
}
```

在這個例子中：

- `try_files $uri $uri/ @backend;` 指示 Nginx 先嘗試匹配 `$uri` 和 `$uri/`，如果都不匹配，則跳轉到 `@backend`。
- 在 `@backend` 位置塊中，添加一個自定義標頭並將請求代理到 `http://backend_server`。

## 使用 `error_page` 與命名位置

假設你希望在遇到 404 錯誤時顯示自定義的錯誤頁面，並且如果特定條件下（例如 URL 中包含 `/api/`），將請求代理到其他服務器。可以這樣配置：

```
server {
    listen 80;
    server_name example.com;

    # 主位置塊
    location / {
        # 通常的處理邏輯
        try_files $uri $uri/ =404;
    }

    # 錯誤頁面處理
    error_page 404 /404.html;
    location = /404.html {
        # 內部重定向到 @notfound 位置
        try_files @notfound;
    }

    # 定義 @notfound 內部重定向位置
    location @notfound {
        # 檢查 URL 中是否包含 /api/
        if ($request_uri ~* /api/) {
            # 如果是，代理到 API 服務器
            proxy_pass http://api_server;
        }

        # 否則返回自定義的 404 頁面
        root /usr/share/nginx/html;
        internal; # 表示這個位置只能內部訪問
    }
}
```

---

🕒修訂版本 #5

★由 treeman 建立於 5 🎮G🎮🎮 2023 01:46:33

✎由 treeman 更新於 1 🎮C🎮🎮 2024 17:23:41