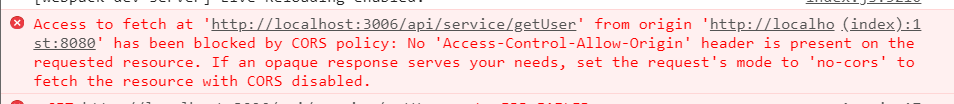
# 跨域问题

## 什么是同源策略

[浏览器的同源策略 - mdn web doc](https://developer.mozilla.org/zh-CN/docs/Web/Security/Same-origin_policy)

如果两个 URL 的 [protocol](https://developer.mozilla.org/zh-CN/docs/Glossary/Protocol)、[host](https://developer.mozilla.org/zh-CN/docs/Glossary/Host)、  [port (en-US)](https://developer.mozilla.org/en-US/docs/Glossary/Port)都相同的话，则这两个 URL 是同源。这个方案也被称为“协议/主机/端口元组”，或者直接是 “元组”。

否则则会出现跨域问题:



## 为什么要理解跨域问题

前端演化进程：

1. Html+css+jquery；可以前后端直接在一个容器中调试开发
2. Webpack打包，调试必须启动前端新端口 webpack-dev-server; 那么和后端一定存在跨域了。



## 存在场景

1. Npm start 本地调试
2. 应用a的前端页面直接访问应用b的后端接口

/opr-platform-srv的前端访问 be-opr-srv的后端接口

1. Iframe中

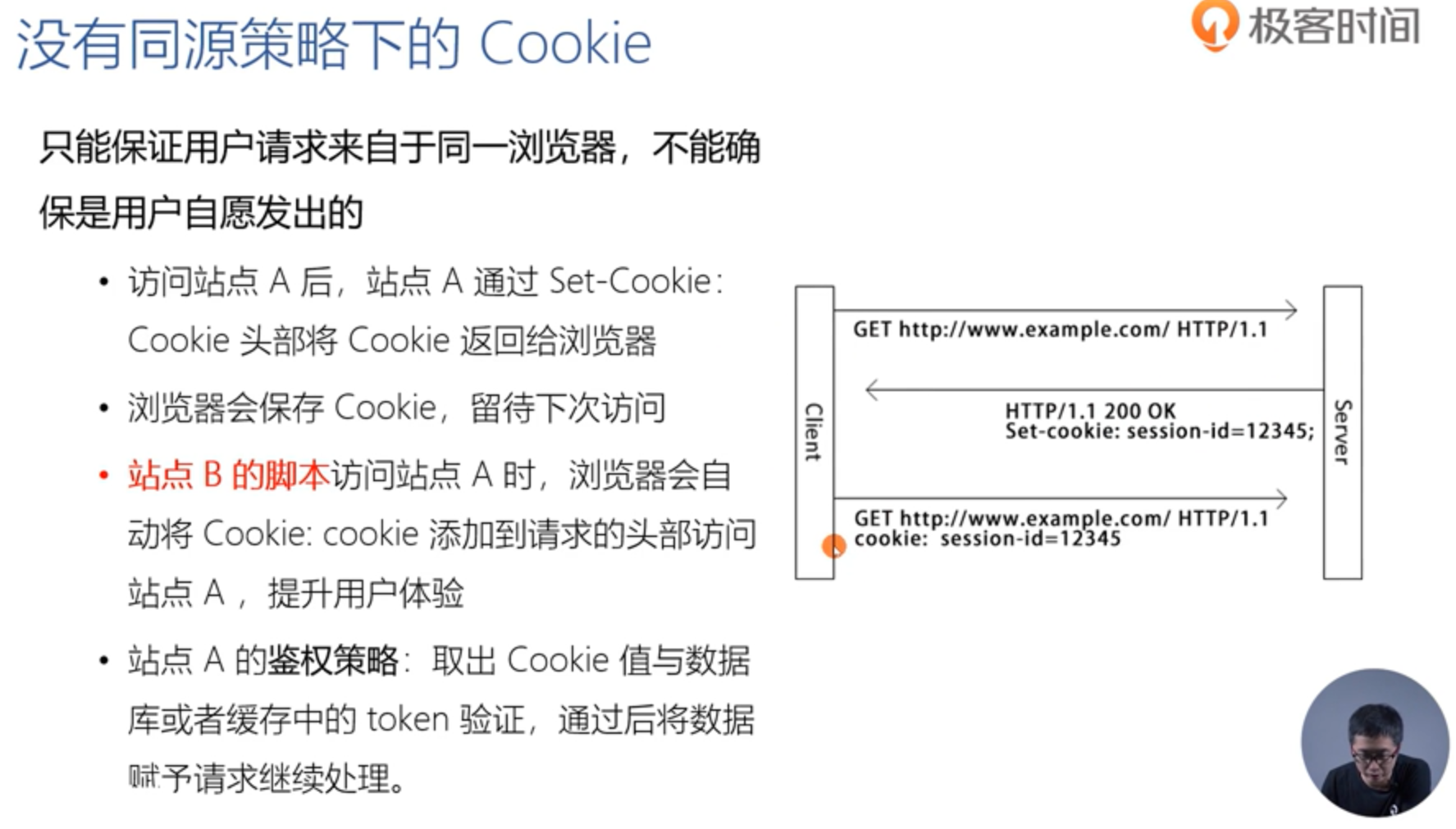
## 跨域问题存在的必要性

[浏览器为什么要有同源策略？- 陶辉 - 极客时间](https://time.geekbang.org/course/detail/100026801-98912)

假设没有同源策略：

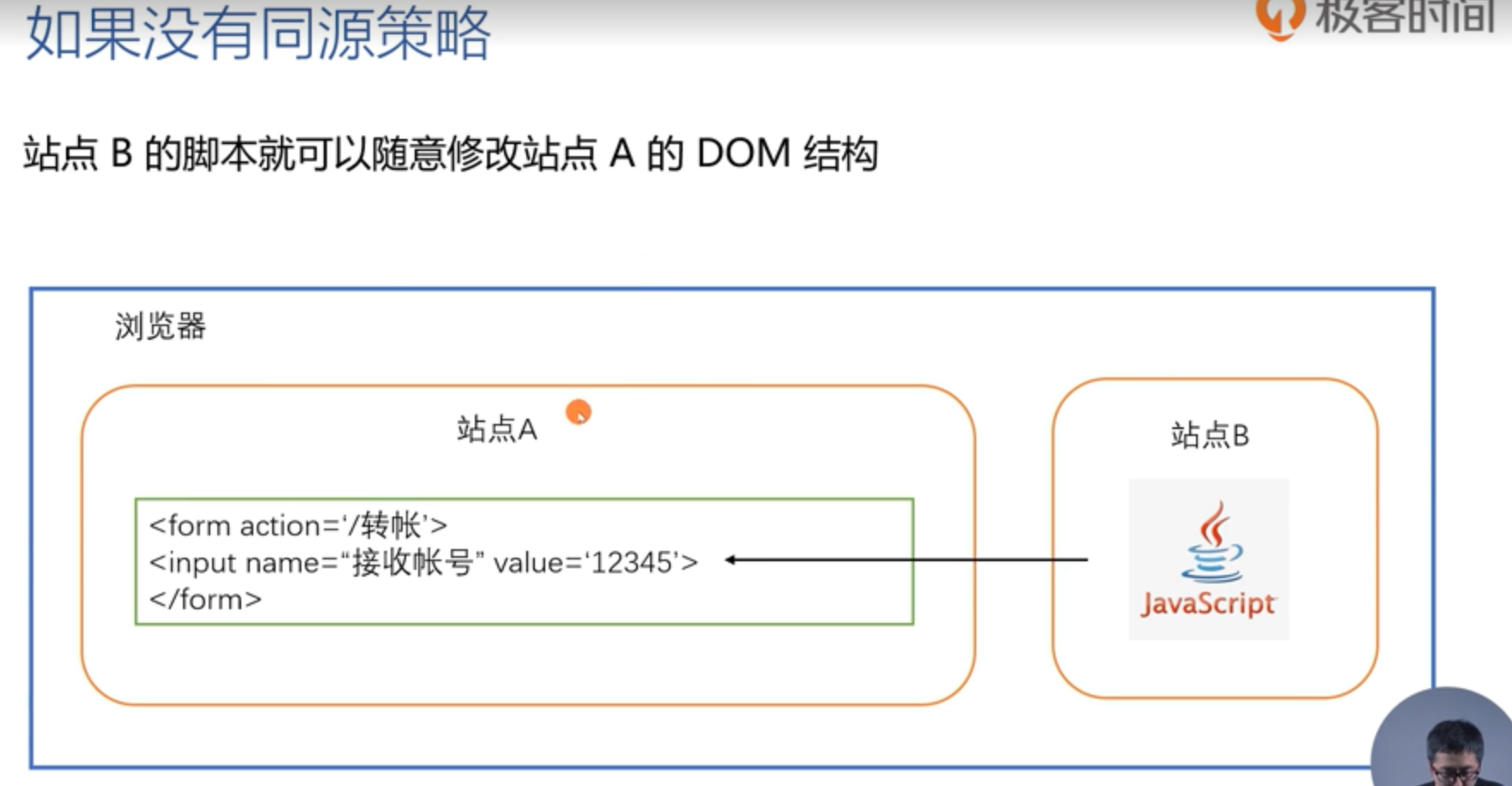
1. 获取其它站点的cookie

且发送Ajax请求，那就可以盗取被人的信息了



1. 更改其它页面的dom元素

即便没有同源策略，这个是否只能存在于iframe中，不同标签页怎么改？



同源策略确实很烦呀，所以：  


疑问：

如果只限制一二，不限制三（Ajax）呢？

浏览器对于不同domain，不给cookie不就好了吗，为啥还要限制ajax请求呢？搞得很麻烦呀；既然后端都可以请求数据，那么就是说明没有cookie的访问是可行的。

# 怎么解决跨域问题



## Jsonp

## 后端允许跨域

CORS: Cross-Origin Resource Sharing

## 反向代理

1. 后端反向代理
2. 前端webpack反向代理
3. Nginx

[反向代理和正向代理区别 - 行为掩饰后超越了思维 - cnblogs](https://www.cnblogs.com/taostaryu/p/10547132.html)

# Nginx拓展

## 基础概念

### 基本操作

[nginx官网](http://nginx.org/en/download.html)

# start nginx

# nginx -s stop

# nginx -s reload

### 配置文件nginx.conf

* 全局块

Nginx全局的一些基本配置

例如并发数：

worker\_processes  1;

* Event块

例如与客户端的最大网络连接数：

events {

    worker\_connections  1024;

}

* Http块

1. http全局块

全局的一些配置，文件引入，超时连接之类的。

    sendfile        on;

    keepalive\_timeout  65;

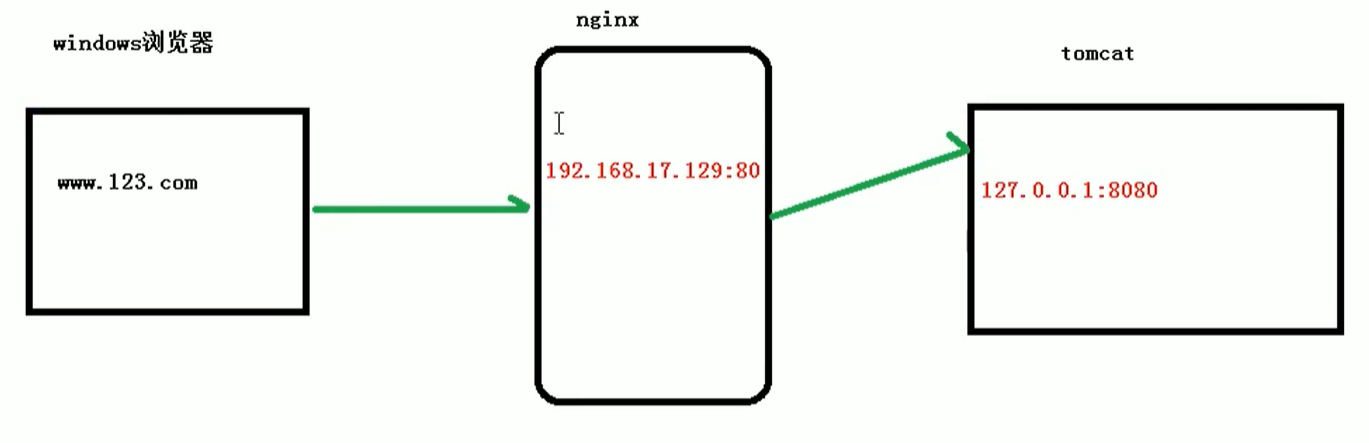
1. server块

一个配置文件可以有多个server块

1. server全局块  
   服务主机，监听端口等
2. location  
   核心部分，路由匹配

## 主要功能

### 反向代理



1. 实现效果  
   浏览器输入 [www.666.com](http://www.666.com) ; 跳转到我们启动的后端服务 localhost:3006
2. 实现效果

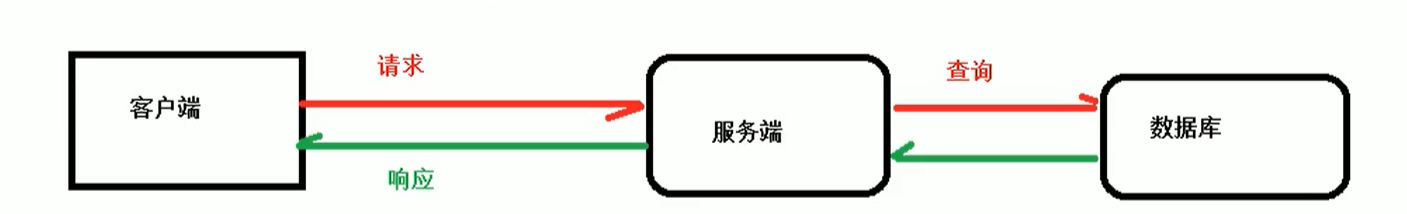
浏览器输入不同的url路径，访问不同的服务器

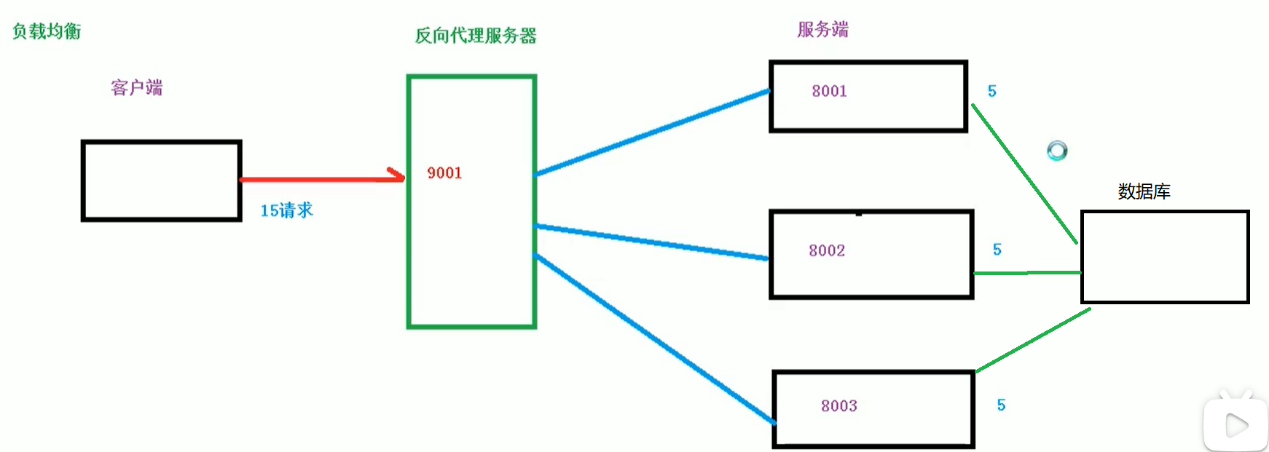
127.0.0.1:8006/user -> http://localhost:3006

127.0.0.1:8006/skill -> http://localhost:3007

### 负载均衡

单个服务请求:

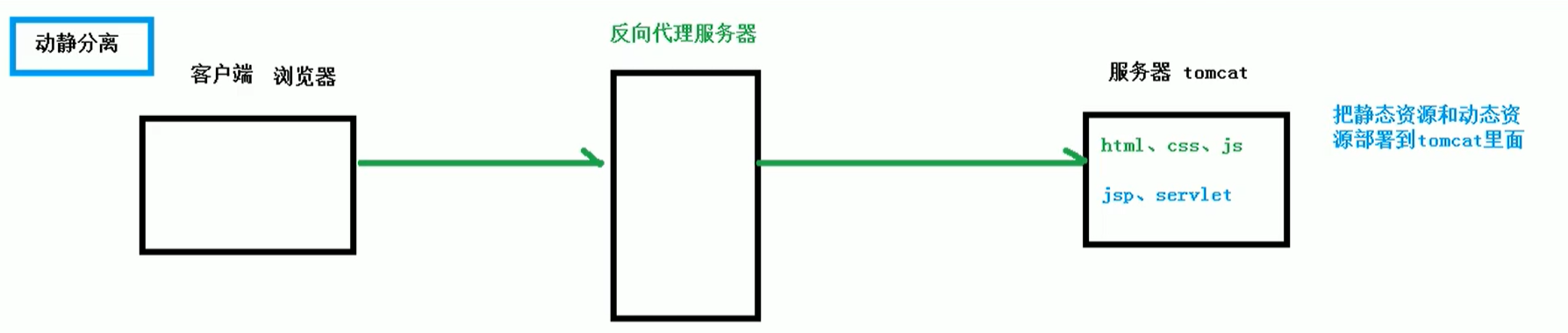


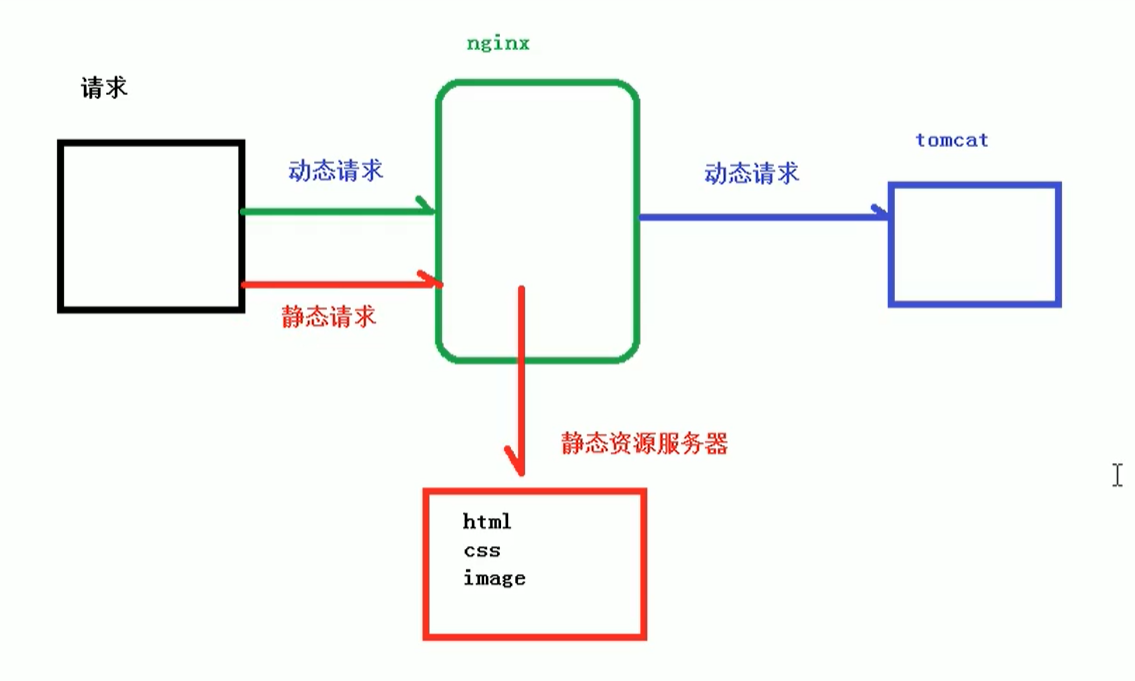


分配策略：

1. 轮询  
   每个请求按照时间顺序逐一分配，能自动剔除宕机的服务
2. Weigth
3. Ip\_hash  
   每个请求按访问ip的hash结果分配，这样每个访客固定访问一个后端服务器，可以解决session问题。
4. Fair（第三方）  
   按后端服务器响应时间来分配

### 动静分离





动静分离：把动态、静态请求分开，并不是单纯的把动态页面、静态页面物理分开。

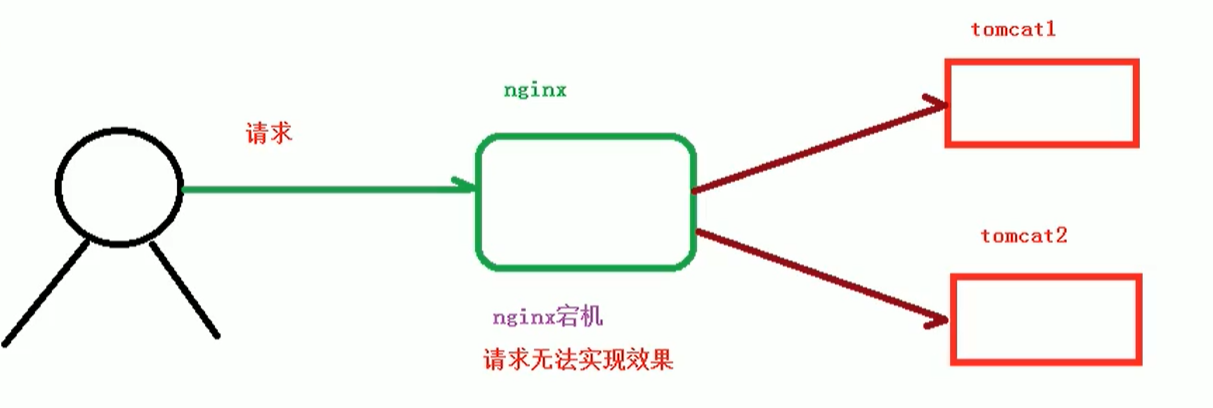
Nginx处理静态页面，tomcat处理动态页面。

从实现角度分为两种：

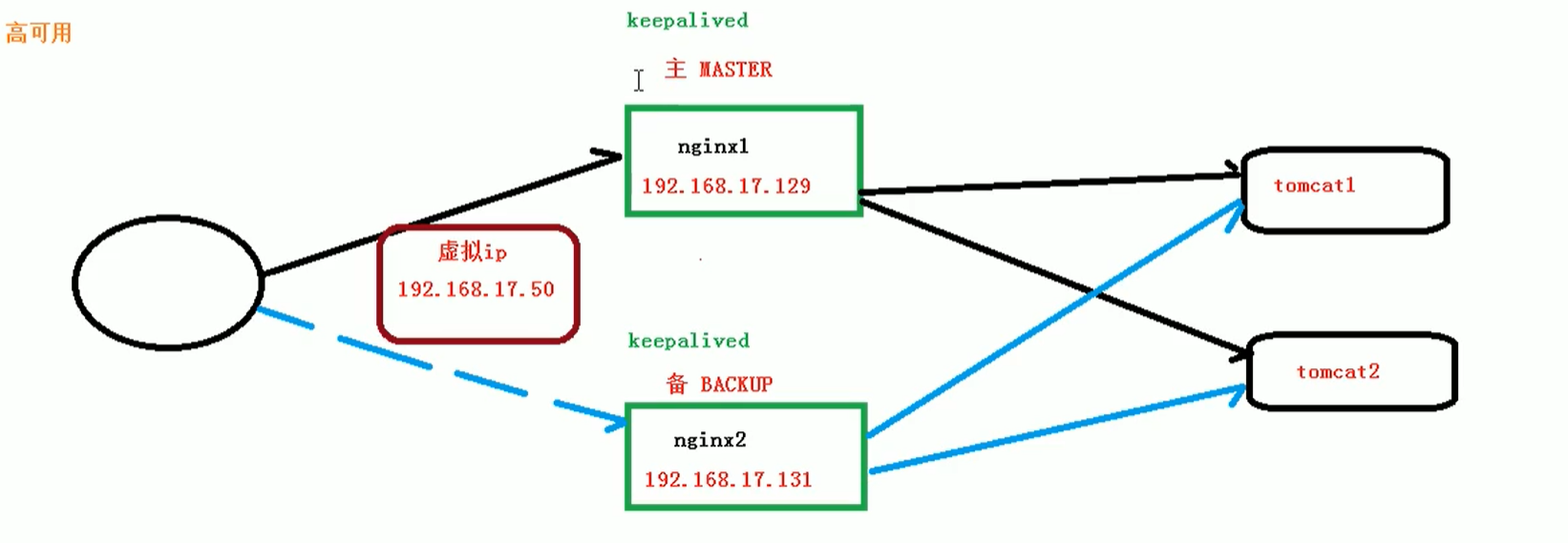
1. 把静态文件分开，部署在另外一台服务器上；
2. 静态文件，后端接口混合部署，通过nginx来区分转发。

针对静态资源，通过location来筛选不同的文件后缀，expires设置文件在浏览器内的过期时间。

## 高可用配置



Nginx宕机了依旧可用，配置一台备份nginx服务器



需要按照keepalived, 然后虚拟出一个ip对外，

问题是：keepalived如果还没nginx稳定呢，挂的次数更多呢？

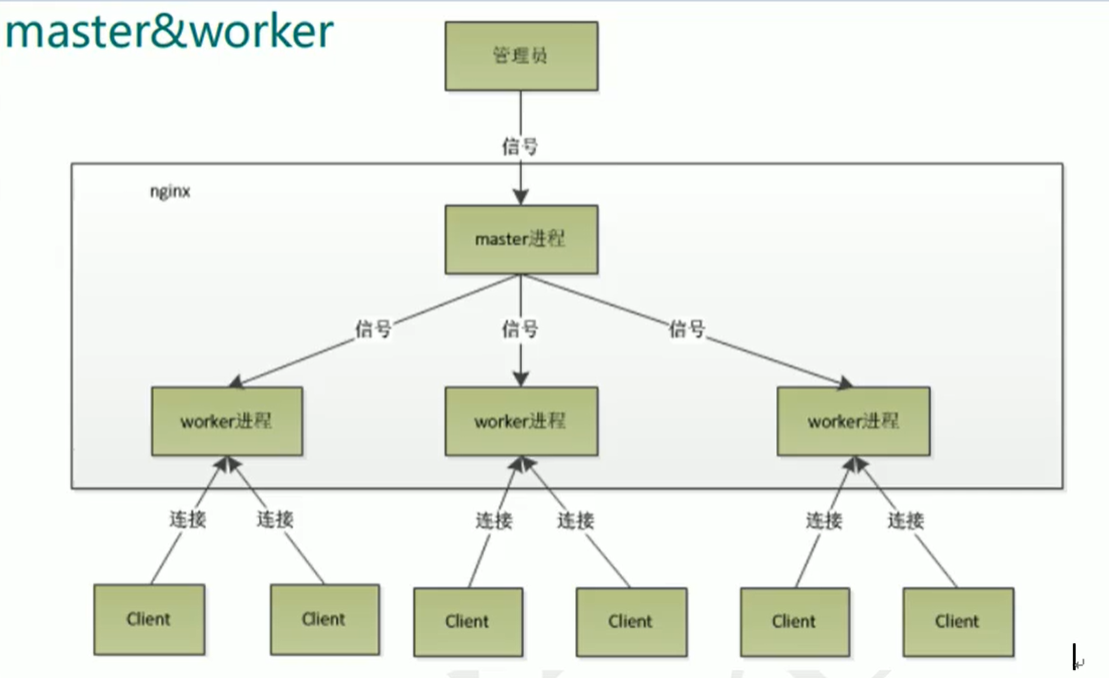
<https://www.keepalived.org/>

貌似只有linux版本？

## Nginx原理

[nginx原理解析 - B站视频](https://www.bilibili.com/video/BV1zJ411w7SV?p=17)

不过感觉老师没讲好，因为我听不懂呢。。。



一个master，多个workers, 好处：

1. 利于nginx -s reload
2. Worker每个都是独立进程，一个挂了，其它的不会中断；

连接数work\_connections

# 参考链接

[【跨域】好好讲讲跨域，以及全部解决方案 - IT楠老师 - B站视频](https://www.bilibili.com/video/BV1nU4y1W7Rf/?spm_id_from=333.788.recommend_more_video.1)

[【狂神说】Nginx最新教程通俗易懂，40分钟搞定！- 遇见狂神说 -B站](https://www.bilibili.com/video/BV1F5411J7vK?from=search&seid=3878820334177914847)

[尚硅谷Nginx教程由浅入深（一套打通丨初学者也可掌握）- Bilibili](https://www.bilibili.com/video/BV1zJ411w7SV?p=13)