Session的工作原理

# Session工作原理

## Session工作原理

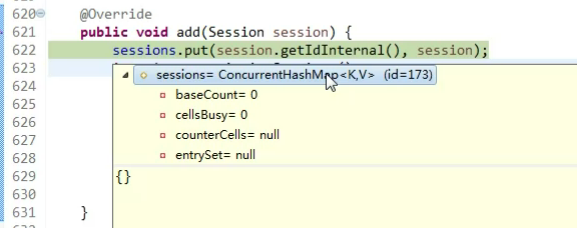
 

## 问题

### **问题1. Tomcat中所有的session都保存在哪里**？（通过源码解读可以查看）

所有的session保存在Tomcat的本地的ConcurrentHashMap(本地缓存)中，以sessionid为key。

源码解读：



### 问题2：Tomcat是怎么追踪到请求是属于哪个session的？

通过cookie：产生会话时，向浏览器发送存有**sessionid的cookie**，后续请求都带上这个cookie。

### 问题3：Session是不是在用户登陆时才能产生？

不是。会话时用来跟踪多个请求的，登陆只是明确会话的主人是谁(把用户信息存放到session中)。

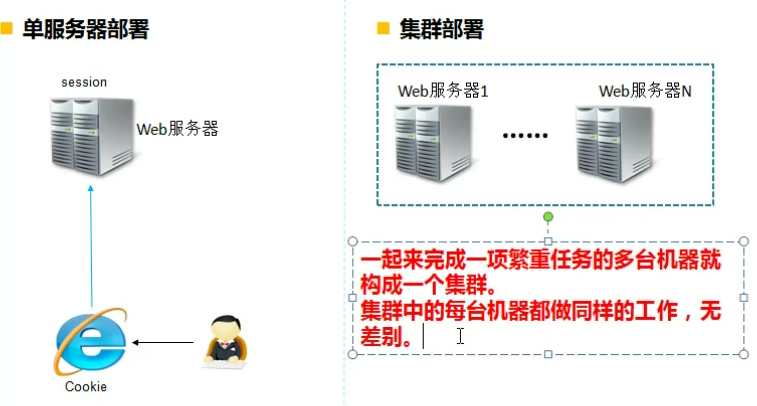
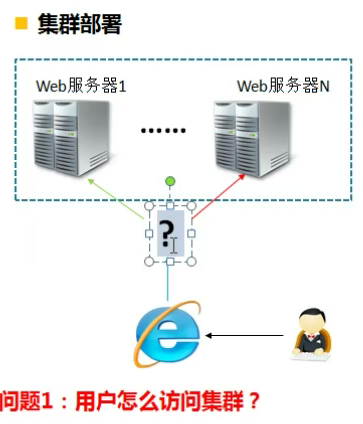
# 集群架构那些事

## 架构师该考虑的问题



集群是最优的选择。

## 集群

# 反向代理-负载均衡(Session集群)

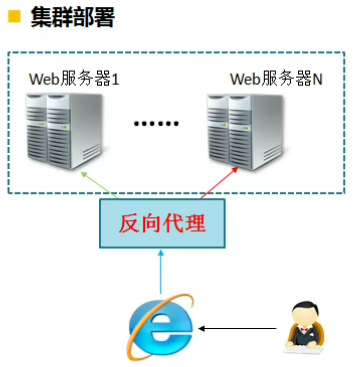
## 反向代理

### 为集群找个代理，代理的职责：

接收用户请求；再分发给集群的机器处理。

### 对这个代理的要求:

并发负载能力强，能均衡分发请求。



## 反向代理-负载均衡的可选方式

### 硬件：

常见的有NetScaler、F5、Radware和Array等；

价格比较昂贵，但提供了高可用性和高稳定性，同时还提供专业的技术服务。

没有专业IT团队的大企业采用。

### 软件

流行的有LVS、haproxy、nginx

三种软件负载均衡器都有开源软件，小企业和互联网公司必选。

## 对于web服务集群—采用nginx

简单，耗资源少；

负载能力强，轻松应对十万并发。



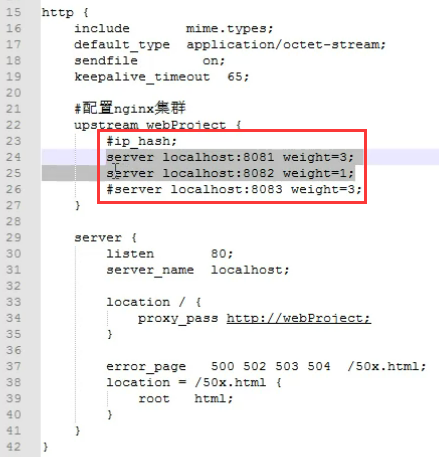
## nginx负载均衡策略

轮询、weight权重、ip\_hash、支持扩展自定义策略。

### 轮询



### weight权重

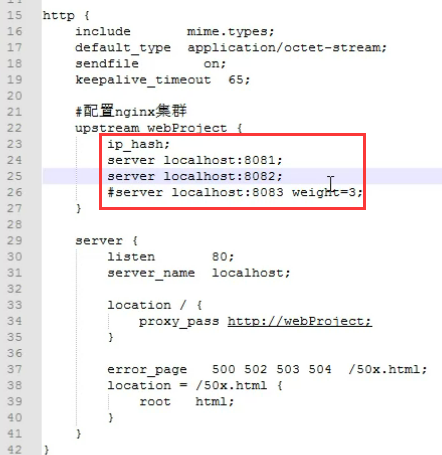


### ip\_hash

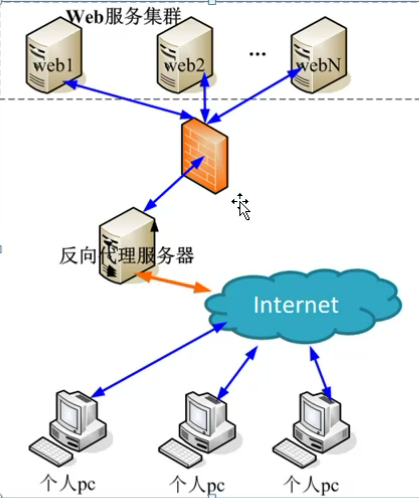
ip\_hash原理：

考虑到在任何一个网络请求中，都有一个源地址和目标地址(源IP和目标IP)。这样，在负载均衡器中，我们就可以利用这两个IP，通过一种**散列算法**把请求分配到不同的服务器上。这种算法就是**目标散列调度(利用目标IP)和源地址散列调度(利用源IP)。**这两种算法为静态算法。

源地址散列调度和目标散列调度属于两种静态的调度算法，在实际应用中，这两种调度算法可以结合使用在防火墙集群中，它们可以保证整个系统的唯一出入口。



# 为什么叫反向代理？



正向代理和反向代理的区别是什么？

正向代理是代理客户端(个人PC)访问外网；

反向代理是代理服务器对外提供服务。

