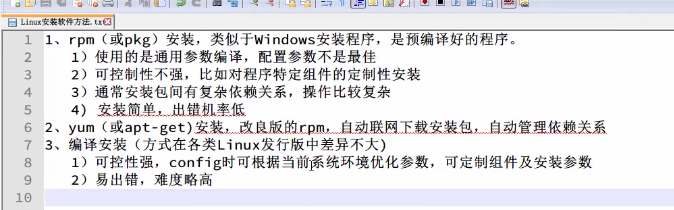
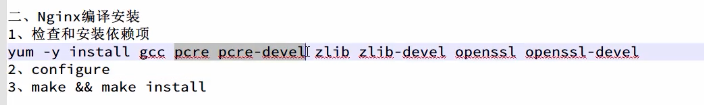
Nginx：方向代理服务器

Linux下安装文件：





Nginx编译安装（使用源码安装）



yum: 自动安装缺失的软件， 如果已经存在就不再安装。 自动配置软件的依赖关系

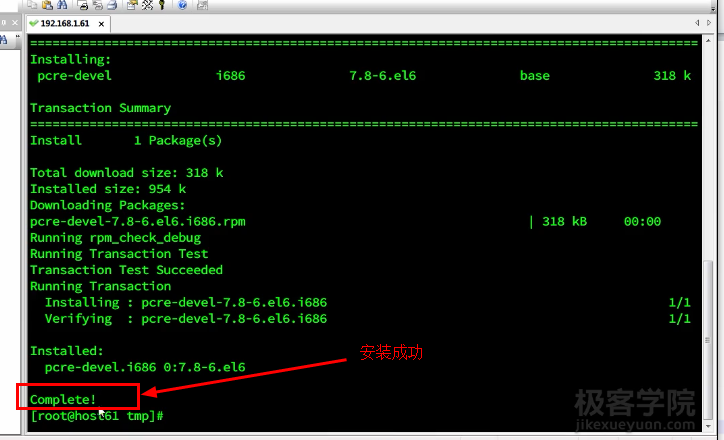
-y: 静默安装， 不提示“是否”

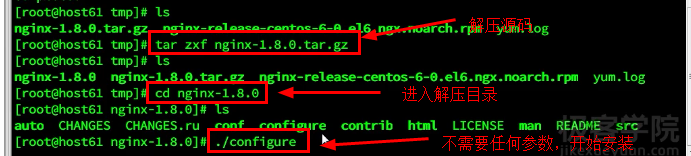
gcc: 编译器

pcr pcr-devel: 执行第二步configure是需要使用正则表达式

zlib zlib-devel: Nginx中传输内容压缩

openssl openssl-devel: 开启https支持





如果需要定制化可以通过 ./configure --help查看可使用的参数

将Nginx配置到/opt/nginx目录下； 如果没有error信息，表示配置成功



make：编译

&&: 两个命令拼接使用

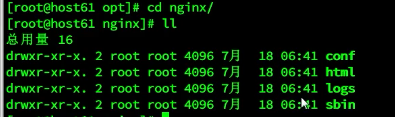
make install: 将编译好的文件复制到指定目录



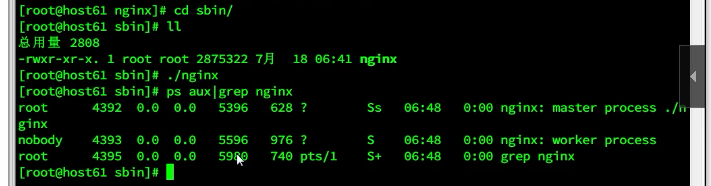
Conf：配置的位置；

HTML：网站的位置;

Sbin:命令的位置



启动Nginx



启动两个进程master进程（主进程， 守护进程）和worker进程（完成响应请求的工作进程）

关闭Nginx -s 操作Nginx



修改参数后， 重新加载参数： ./nginx –s reload

配置Nginx随系统的启动而启动。 找到系统的自启动脚本， 将nginx的启动命令添加到该文件即可

Nginx的配置nginx.conf

Nginx是基于模块设计的一个应用， 所以其配置文件也是按照模块划分的。

Worker\_processes 1; 顶层配置信息管理服务器级别行为

event指令与事件模型有关，

Events

#配置处理 可以响应连接数量

Worker\_connections 1024;

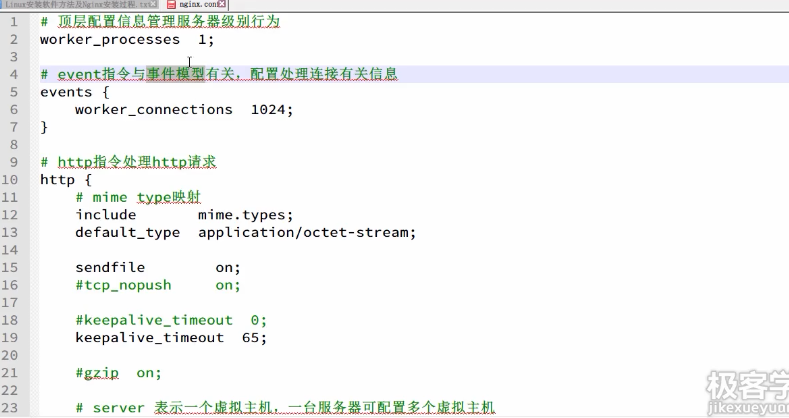
http: #http指令处理http请求

include： 引入另外一个配置文件

include mime.types： mime的type映射文件， 响应的mime类型。 如html类型

default\_type：当类型不能在mime.type中匹配时使用该默认的mime类型

default\_type application/octet-stream 默认使用二进制流模式



一个http下面可以有若干个server； server\_name用来匹配请求域名的。 如果都匹配不到， 则使用默认配置的server， 默认server为第一个配置的server。 一个server下可以有若干个location

Location下的proxy\_pass是将匹配到的location再转发到指定的位置

Location语法

“=”： 精确匹配

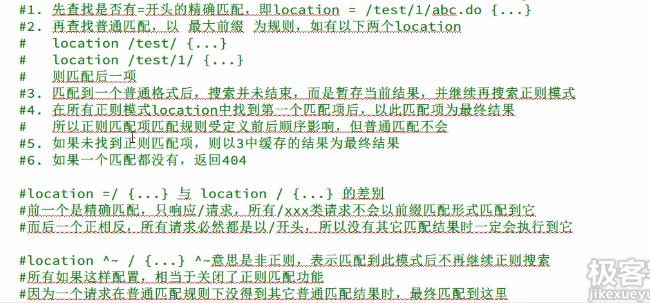
“~”:正则匹配

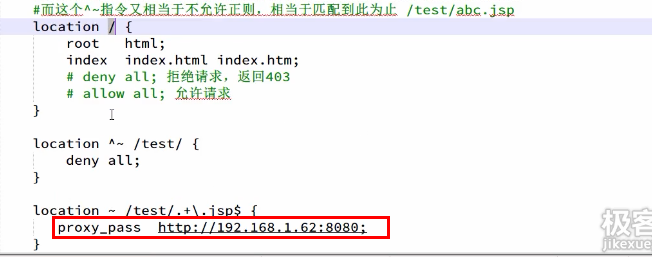
“~\*”：正则匹配， 不区分大小写

“^~”:关闭正则匹配

“@”： 类似于变量定义。 错误转发到其他网站时， 不允许直接转发。 可以使用该值。



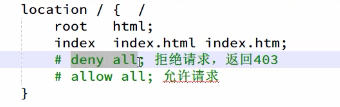




处理行为：  
 1. 再本地找静态文件， 将其响应给浏览器

2. 将请求转发到其他服务器， 将其他服务的响应返回给浏览器。

静态响应：



root： 静态资源的位置，可以是相对路径也可以是绝对录了。 相对路径以Nginx安装目录的html目录开始

index：默认的请求页面， 当请求地址为根目录时， 默认按照index配置的文件名响应文件。

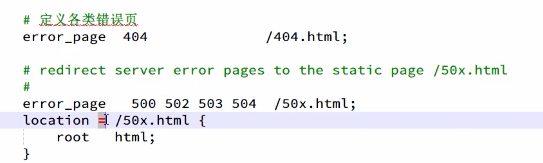
Location匹配规则：1.精确匹配； 2.正则匹配

1. 精确匹配

Location =/test 只匹配/test, /text.html 就匹配不到。

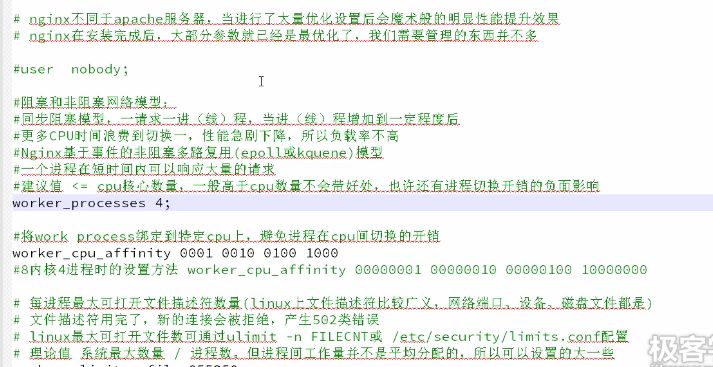
1. 正则匹配

响应错误码，默认对应的页面



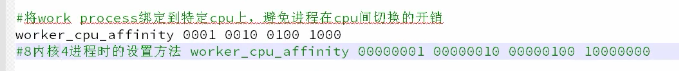


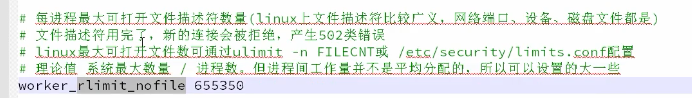
# Nginx优化



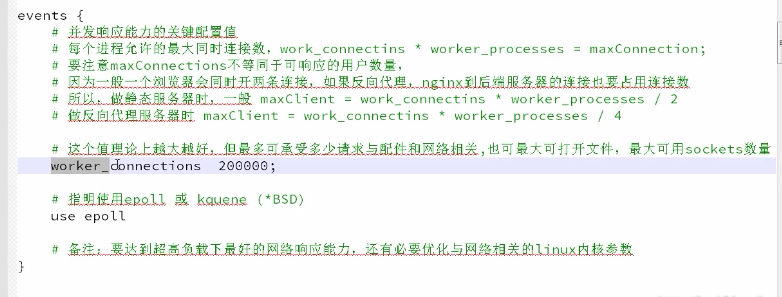
Worker\_processes 建议值<=cpu核心数。 如果进程数太多， 时间会消耗在线程切换上

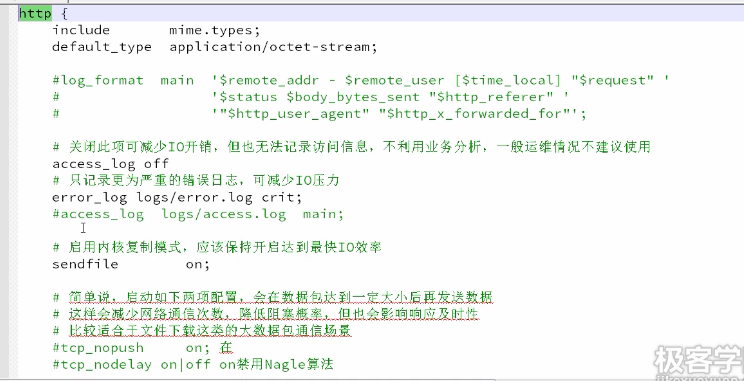
Worker\_cpu\_affinity： 将work\_process绑定到特定的cpu上， 避免进程在cpu间切换的开销。

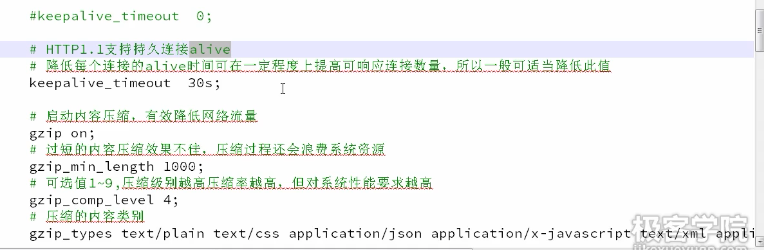




工作进程最大连接数。可以配置Nginx的负责量。









解压文件， 并将解压文件放到指定目录 –C



/etc/profile.d/\*.sh 系统启东时会扫描该目录下的所有.sh文件。

Export 将配置的环境变量导出



重新加载环境变量



Tomcat优化： 1. 内存使用配置 2. 最大连接数

内存使用配置

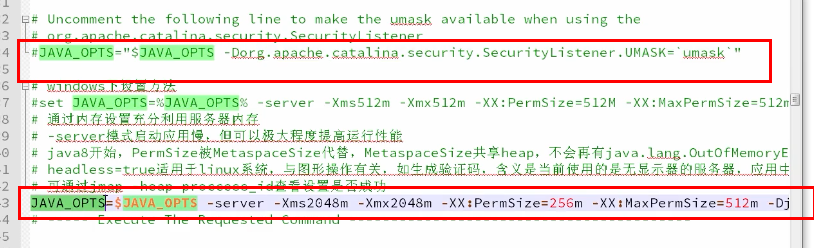
Catalina.sh 设置jvm启动参数

-server: 让Java以server模式运行， 该模式，启动慢，但是运行快

-Xms –Xmx：最小/大堆内存， 最好保持一致。避免运行过程中内存反复扩展。

-XX:PermSize -XX:MaxPermSize:默认类加载内存大小；最大类加载内存 1.8之后取消了该内存模型， 开始使用元空间的概念， 元空间和堆内存共享

-Djava.awt.headless=true: 设置为无图形显示模式， 调用绘图api时，不会报错



Protocol：设置Tomcat的当前网络模型（bio， nio, apr:系统级异步io，大文件处理时，效果明显）

maxThreads:最大可使用线程数量

minSpareThreads：最小空闲线程数量

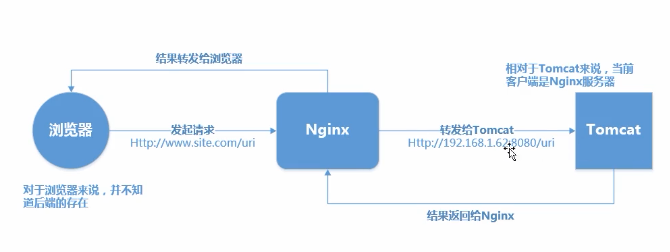
maxSpareThread：最大空闲线程数量

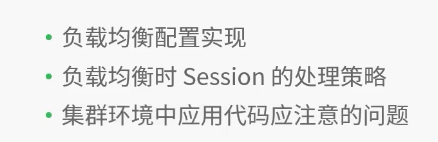
acceptCount：当前处理数量达到最大线程数量时，允许在接受请求的数量， 当有线程空闲时，就调用空闲线程。

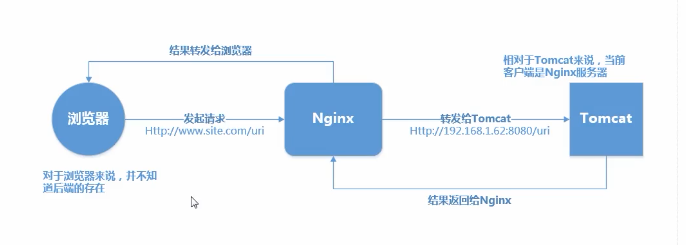
总的可接受线程数量= maxThreads + acceptCount； 如果超过了该数量， 则不会再接受新的请求而报错。

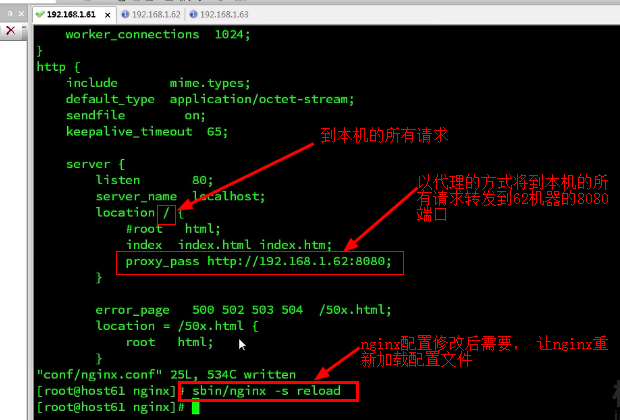
enableLookups：dns反查客户机主机名

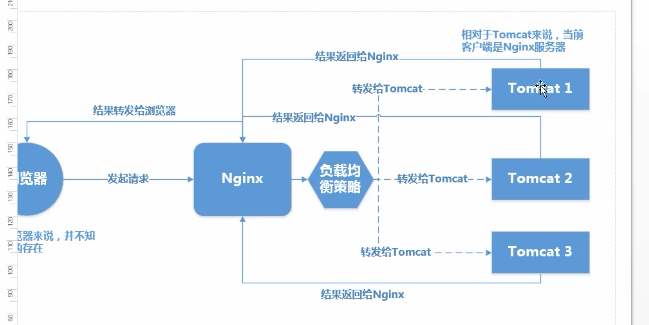






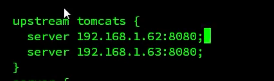


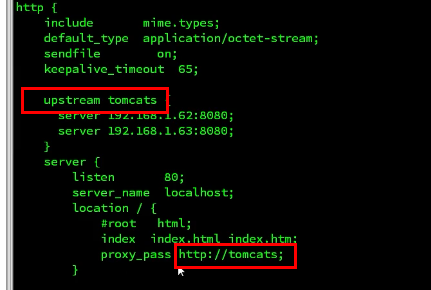




服务器组：

http—>Upstream 只需要设置ip即可， 不需要协议； 默认为平均策略。





backup:备份机器， 当所有非备份机全部挂掉后， 请求才会发到该机器

down：停机标志， 请求永远不会发到该机器。 作用：当需要维护集群中的某台机器时， 可以在该sever后面添加down停机标志， 这样Nginx的请求就不会再发到该机器上。 如果直接停止后台服务， Nginx还会尝试将请求发送到该机器， 造成资源浪费（该机器指定的服务已经停止工作了。）

负载均衡策略：

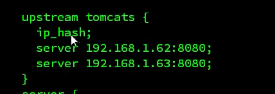
自带负载均衡策略：

None:平均策略

Weight：默认为1， 加权轮询

Ip\_hash: 将客户的ip通过hash算法，然后绑定到某台服务器， 除非客户的ip或者服务器断掉。 否则， 该ip的可以一直连接这台服务器。

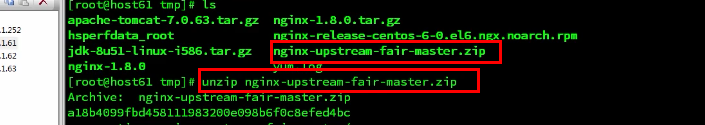
使用ip\_hash算法



第三方扩展的策略：

Fair：根据服务器性能，动态分布请求数量

url\_hash：类似于ip\_hash，将url绑定到指定服务

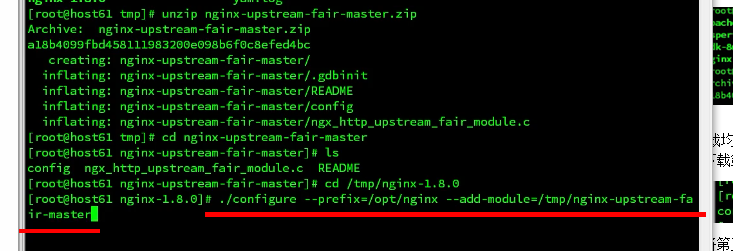


第三方负载均衡策略，安装配置

1. 下载第三方文件，解压文件， “.c”文件是该第三方软件的源码。



1. 将第三方软件编译安装到Nginx； 执行Nginx的configure脚本， 编译Nginx， 指定Nginx的位置， 同时显示指定编译需要加载的第三方软件。 目录即可



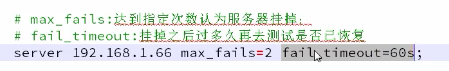
1. 如果是首次安装Nginx， 执行configure后， 可以执行mask && make install。 如果已经安装过Nginx， 此时不能在使用make&& make install. 这样会覆盖原有的安装。 只需要执行make命令即可。 仅仅是编译， 编译的文件还在当前位置， 这样并不会替换目标文件。
2. 编译结果位置 当前编译位置目录下的objs目录下， 会有一个新生成的nginx可执行文件， 该文件就是携带第三方软件的编译后文件。
3. 用新编译的nginx执行文件替换原来的执行文件。 先停掉原来的nginx服务。 在使用该nginx覆盖原有的执行文件。



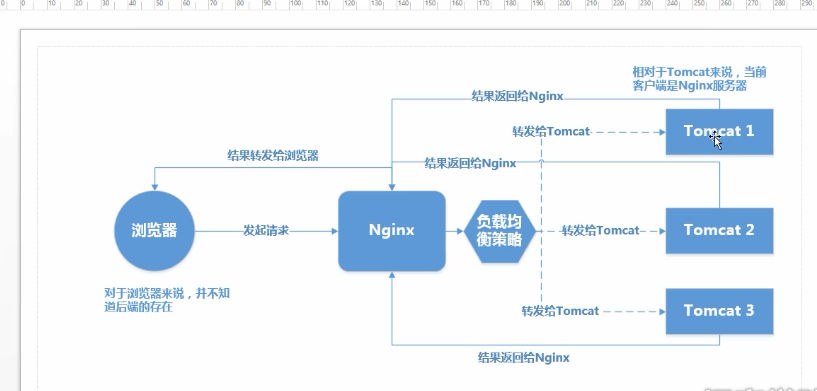


max\_fails: 请求到达该机器响应失败次数达到指定值时才能为该机器宕机， 不在发送请求到该机器。

fail\_timeout：当认为该机器宕机后， 多长时间后尝试测试该机器是否恢复服务。



# 负载均衡是session



1. 将用户锁定到指定服务器， 粘性session。 可以使用ip\_hash均衡策略。 简单，不需要对session做处理。 容错性不好。当一台服务器宕机后， 用户请求被分发到其他服务器， 此时用户的登录等session状态丢失。 问题严重性低，
2. Session复制。一台服务器的session发生任何变动， 都将其变动广播到整个集群服务器中。 是集群中的服务器保证session同步。 提高其容错性。但一处的变动需要修改整个集群， 提升了网络的负荷压力。 性能低下，甚至引起网络故障。

Tomcat配置集群， 将engine中的Cluster注释去掉即可

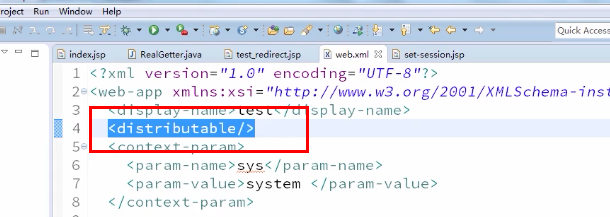


默认的配置不一定都满足我们的需求， 有时候需要我们手动设置一些值。 如当多个Tomcat实例放在同一台服务器时，端口不能冲突。

一般服务器都有两块网卡， 一块对内， 一块对外。 集群统一使用对内的网卡， 如果有的设置到对外的网卡上， 则Tomcat不能进行通信， 就不能完成session复制。

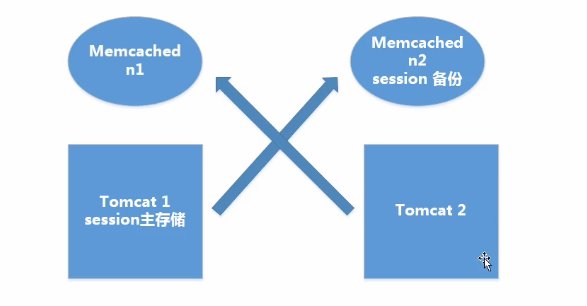
Address设置Tomcat绑定的网卡ip

应用中的配置：该配置表示， 当前应用可在集群下运行。 也就可以实现session复制功能了



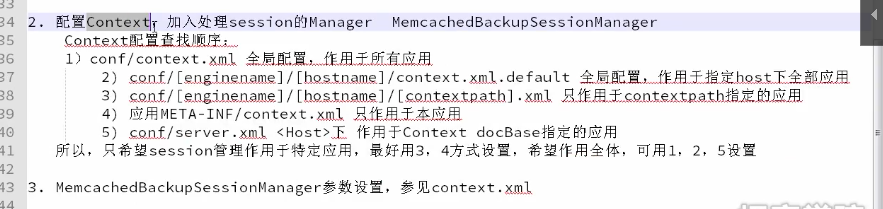
1. 创建共享session， 服务器使用时，去共享处获取。

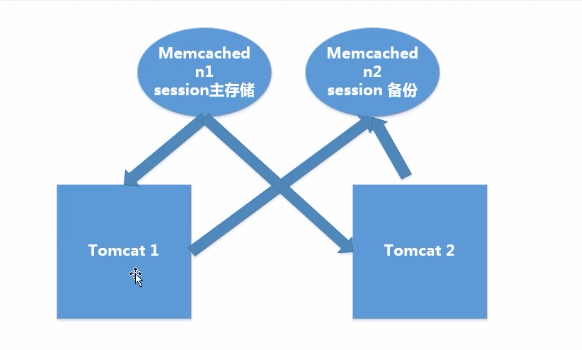
粘性：主要存放在Tomcat中， 如果Tomcat宕机， 则从memcached获取，并存在本地内存。

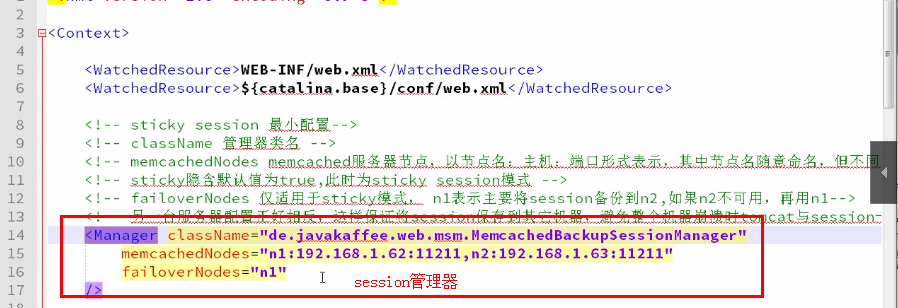


非粘性： Tomcat不在存储session

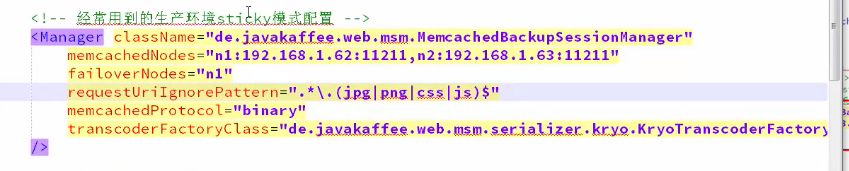
配置方式：





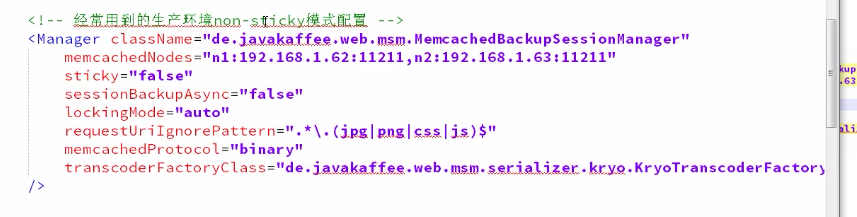


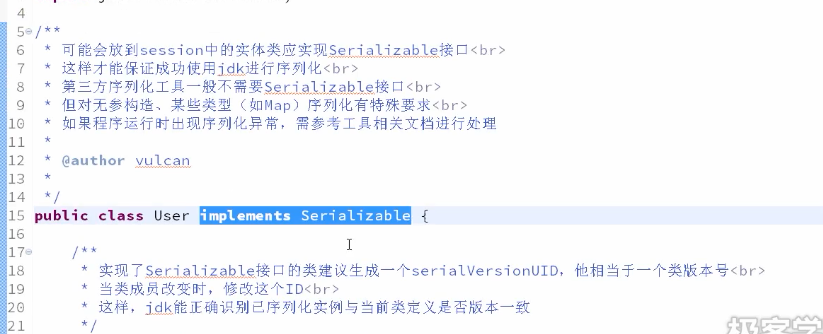
粘性：

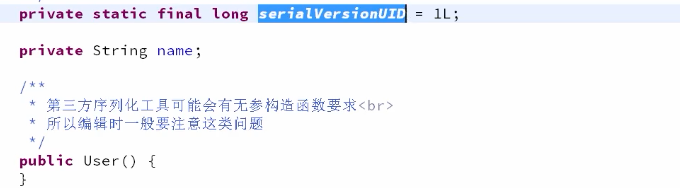


非粘性：

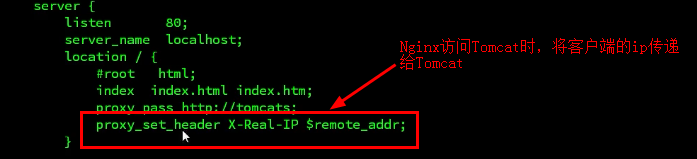
lockingMode和sessionBackupAsync保证数据的一致性

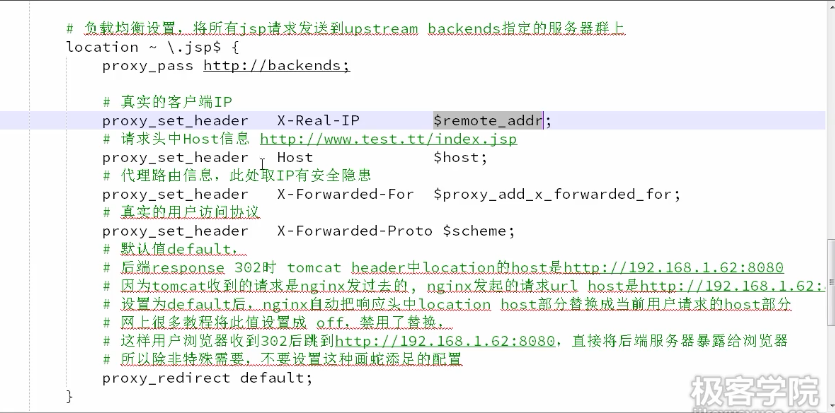






进过Nginx的代理， Tomcat获取客户端的真实ip， 将真实ip写到请求的header中， 否则获取到的是Nginx的ip







动静分离时， 引用静态资源可使用一个变量承接域名， 然后再跟资源位置拼接。

域名可以配置在web.xml中

