1.下面先做最简单的配置，意思就是把请求到本机8080端口的所有http请求分别转发到本机18080和28080端口：

1. #user  nobody;
2. worker\_processes  2;
4. #error\_log  logs/error.log;
5. #error\_log  logs/error.log  notice;
6. #error\_log  logs/error.log  info;
8. #pid        logs/nginx.pid;

11. events {
12. #使用网络IO模型linux建议epoll，FreeBSD建议采用kqueue，window下不指定。
13. #use epoll;
14. worker\_connections  1024;
15. }

18. http {
19. include       mime.types;
20. default\_type  application/octet-stream;
22. sendfile        on;
24. keepalive\_timeout  65;
26. #gzip  on;
27. upstream mysvr {
28. #根据ip计算将请求分配各那个后端tomcat，许多人误认为可以解决session问题，其实并不能。
29. #同一机器在多网情况下，路由切换，ip可能不同
30. #ip\_hash;
31. server localhost:18080;
32. server localhost:28080;
33. }
35. server {
36. listen       8080;
37. server\_name  localhost;
39. #charset koi8-r;
41. #access\_log  logs/host.access.log  main;
43. location / {
44. proxy\_connect\_timeout   3;
45. proxy\_send\_timeout      30;
46. proxy\_read\_timeout      30;
47. proxy\_pass http://mysvr;
48. }
49. }
50. }

2.Nginx upstream的5种权重分配方式   
1）、轮询（默认）   
每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器down掉，能自动剔除。   
2）、weight   
指定轮询几率，weight和访问比率成正比，用于后端服务器性能不均的情况。   
例如：   
upstream mysvr {   
server 192.168.0.14:8080 weight=10;   
server 192.168.0.15:8080 weight=10;   
}   
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的问题（在上面提到的，如果一台机器有多个ip，且都能达到我们的服务器，这点就无效了）。   
（适用于要求ip地址一致性的场景）   
例如：   
upstream mysvr {   
ip\_hash;   
server 192.168.0.14:8080;   
server 192.168.0.15:8080;   
}   
4）、fair（第三方）   
按后端服务器的响应时间来分配请求，响应时间短的优先分配。   
（fair策略是扩展策略，默认不被编译进nginx内核。其原理是根据后端服务器的响应时间判断负载情况，从中选出负载最轻的机器进行分流。这种策略具有很强的自适应性，但是实际的网络环境往往不是那么简单，因此要慎用。）   
upstream mysvr {   
server 192.168.0.14:8080;   
server 192.168.0.15:8080;   
fair;   
}   
5）、url\_hash（第三方）   
按访问url的hash结果来分配请求，使每个url定向到同一个后端服务器，后端服务器为缓存时比较有效。（适用于请求一致性，如缓存服务器）   
例：在upstream中加入hash语句，server语句中不能写入weight等其他的参数，hash\_method是使用的hash算法   
upstream mysvr {   
server 192.168.0.14:8080;   
server 192.168.0.15:8080;   
hash $request\_uri;   
hash\_method crc32;   
}   
  
3.定义负载均衡设备的Ip及设备状态   
upstream mysvr {   
ip\_hash;   
server 127.0.0.1:9090 down;   
server 127.0.0.1:8080 weight=2;   
server 127.0.0.1:6060;   
server 127.0.0.1:7070 backup;   
}    
在需要使用负载均衡的server中增加    
proxy\_pass <http://bakend/>;    
   
每个设备的状态设置为:    
1.down 表示单前的server暂时不参与负载    
2.weight 默认为1.weight越大，负载的权重就越大。    
3.max\_fails：允许请求失败的次数默认为1.当超过最大次数时，返回proxy\_next\_upstream 模块定义的错误    
4.fail\_timeout:max\_fails次失败后，暂停的时间。    
5.backup： 其它所有的非backup机器down或者忙的时候，请求backup机器。所以这台机器压力会最轻。