查看Nginx进程：ps -ef | grep Nginx

停止： /usr/local/nginx/sbin/nginx -s stop

启动： /usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf

**安装步骤：**

先安装依赖包

yum install openssl-devel pcre-devel -y

解压nginx包

 tar -zxvf nginx-1.11.13.tar.gz

cd /usr/local/src/nginx-1.11.13

 ./configure

yum install make -y

make && make install

修改配置nginx.cong文件

 vi /usr/local/nginx/conf/nginx.conf

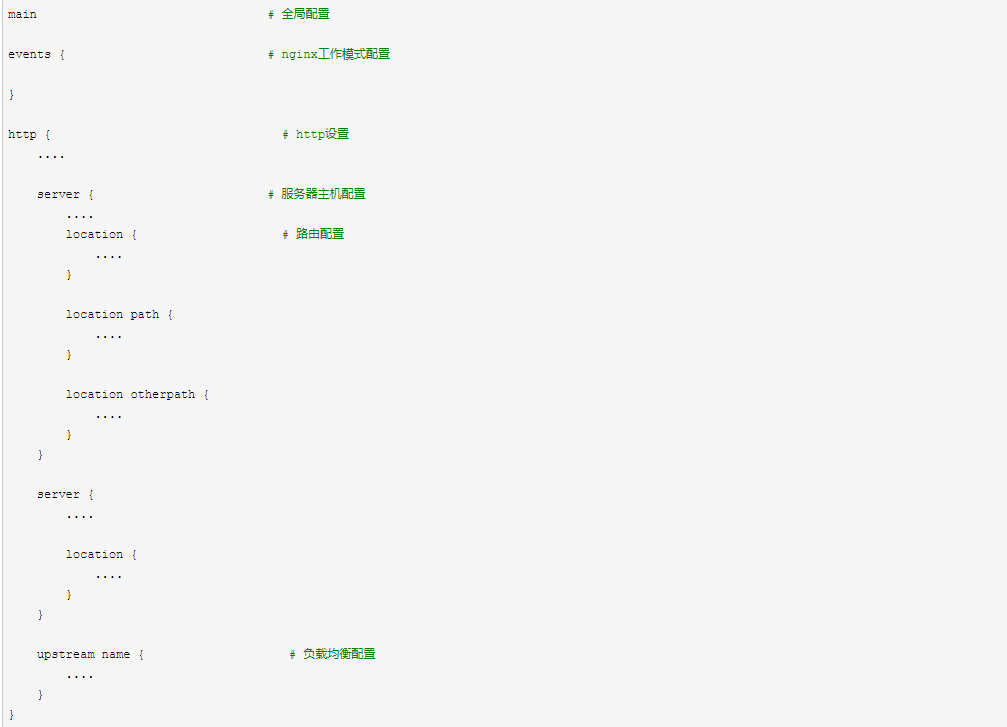
修改tomcat端口

vi /data/tomcat/conf/server.xml

Nginx配置：nginx是一个功能非常强大的web服务器加反向代理服务器，同时又是邮件服务器等等

在项目使用中，使用最多的三个核心功能是反向代理、负载均衡和静态服务器

这三个不同的功能的使用，都跟nginx的配置密切相关，nginx服务器的配置信息主要集中在nginx.conf这个配置文件中，并且所有的可配置选项大致分为以下几个部分



如上述配置文件所示，主要由6个部分组成：

1.main：用于进行nginx全局信息的配置

2.events：用于nginx工作模式的配置

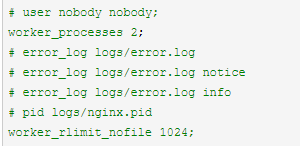
3.http：用于进行http协议信息的一些配置

4.server：用于进行服务器访问信息的配置

5.location：用于进行访问路由的配置

6.upstream：用于进行负载均衡的配置

**main模块**



上述配置都是存放在main全局配置模块中的配置项

user用来指定nginx worker进程运行用户以及用户组，默认nobody账号运行

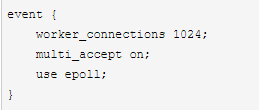
worker\_processes指定nginx要开启的子进程数量，运行过程中监控每个进程消耗内存(一般几M~几十M不等)根据实际情况进行调整，通常数量是CPU内核数量的整数倍

error\_log定义错误日志文件的位置及输出级别【debug / info / notice / warn / error / crit】

pid用来指定进程id的存储文件的位置

worker\_rlimit\_nofile用于指定一个进程可以打开最多文件数量的描述

**event 模块**



上述配置是针对nginx服务器的工作模式的一些操作配置

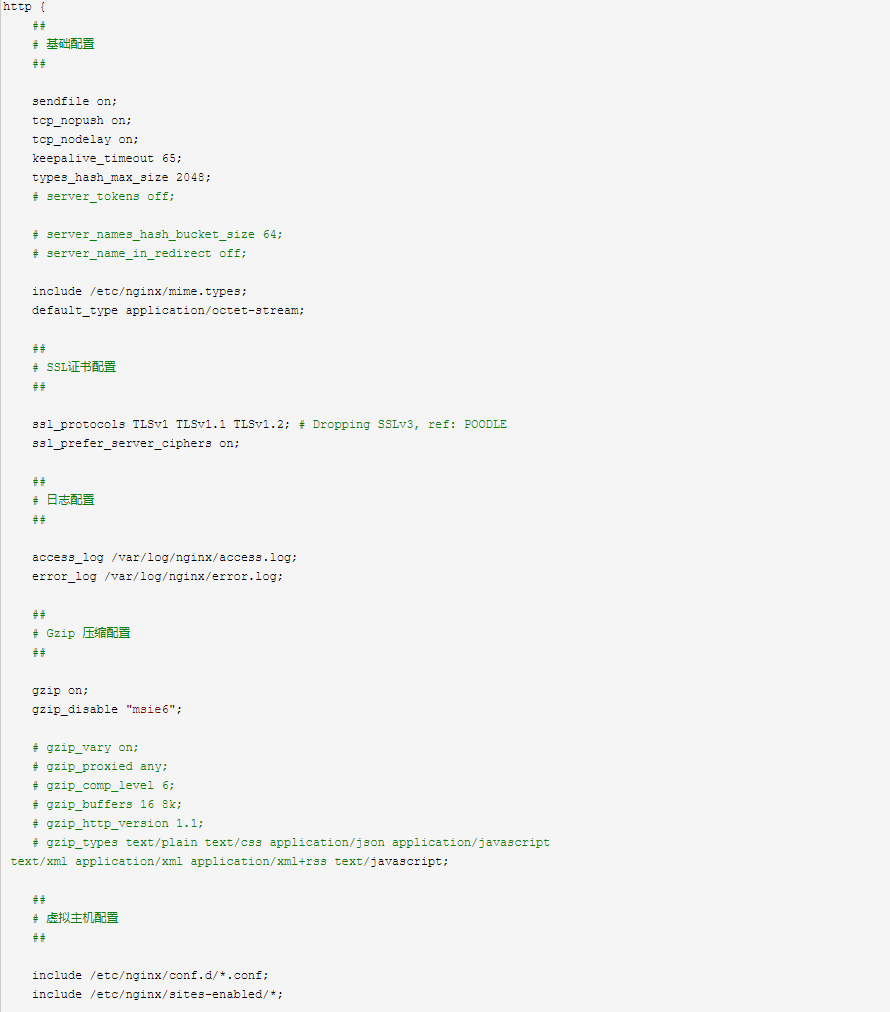
worker\_connections 指定最大可以同时接收的连接数量，这里一定要注意，最大连接数量是和worker processes共同决定的。

multi\_accept 配置指定nginx在收到一个新连接通知后尽可能多的接受更多的连接

use epoll 配置指定了线程轮询的方法，如果是linux2.6+，使用epoll，如果是BSD如Mac请使用Kqueue

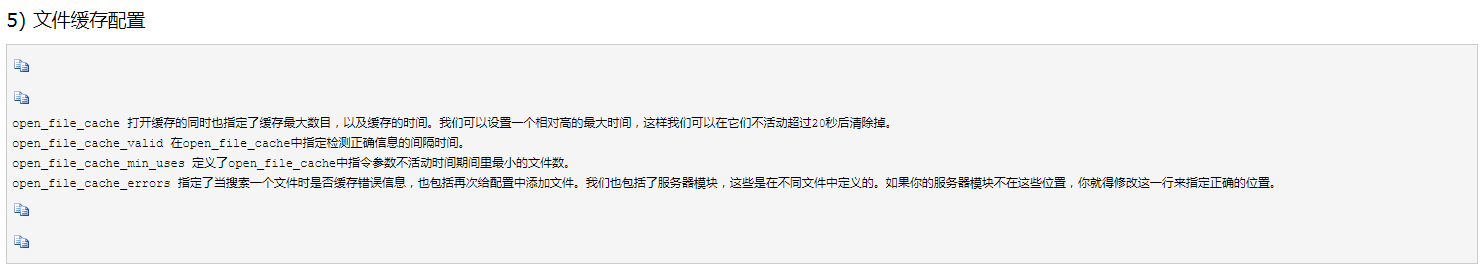
**http模块**

作为web服务器，http模块是nginx最核心的一个模块，配置项也是比较多的，项目中会设置到很多的实际业务场景，需要根据硬件信息进行适当的配置，常规情况下，使用默认配置即可！









**server模块**

srever模块配置是http模块中的一个子模块，用来定义一个虚拟访问主机，也就是一个虚拟服务器的配置信息



核心配置信息如下：

server：一个虚拟主机的配置，一个http中可以配置多个server

server\_name：用力啊指定ip地址或者域名，多个配置之间用空格分隔

root：表示整个server虚拟主机内的根目录，所有当前主机中web项目的根目录

index：用户访问web网站时的全局首页

charset：用于设置www/路径中配置的网页的默认编码格式

access\_log：用于指定该虚拟主机服务器中的访问记录日志存放路径

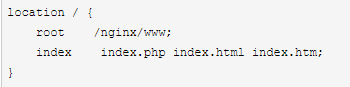
error\_log：用于指定该虚拟主机服务器中访问错误日志的存放路径

**location模块**

location模块是nginx配置中出现最多的一个配置，主要用于配置路由访问信息

在路由访问信息配置中关联到反向代理、负载均衡等等各项功能，所以location模块也是一个非常重要的配置模块

基本配置



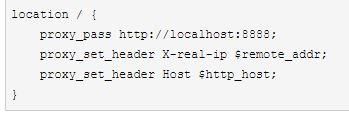
location /：表示匹配访问根目录

root：用于指定访问根目录时，访问虚拟主机的web目录

index：在不指定访问具体资源时，默认展示的资源文件列表

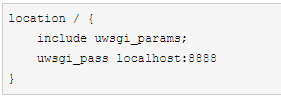
反向代理配置方式

通过反向代理代理服务器访问模式，通过proxy\_set配置让客户端访问透明化



uwsgi配置

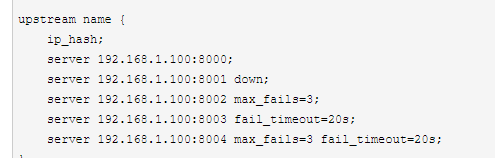
wsgi模式下的服务器配置访问方式



**upstream模块**

upstream模块主要负责负载均衡的配置，通过默认的轮询调度方式来分发请求到后端服务器

简单的配置方式如下



核心配置信息如下

ip\_hash：指定请求调度算法，默认是weight权重轮询调度，可以指定

server host:port：分发服务器的列表配置

-- down：表示该主机暂停服务

-- max\_fails：表示失败最大次数，超过失败最大次数暂停服务

-- fail\_timeout：表示如果请求受理失败，暂停指定的时间之后重新发起请求

反向代理配置

server {

listen 80;

server\_name 172.168.50.201;

#charset koi8-r;

#access\_log logs/host.access.log main;

location ~ /vod/ {

proxy\_pass http://172.168.50.200:8081; //输入172.168.50.201/vod 访问172.168.50.200:8081

index index.html index.htm;

}

location ~ /edu/ {

proxy\_pass http://172.168.50.200:8080; //输入172.168.50.201/edu 访问172.168.50.200:8081

index index.html index.htm;

负载均衡配置

http{

........

upstreat myserver { //自定义名称

server 172.168.50.200:8081;

server 172.168.50.200:8080; //服务器列表

}

server {

listen 80;

server\_name 172.168.50.201;

#charset koi8-r;

#access\_log logs/host.access.log main;

location ~ /vod/ {

proxy\_pass http://myserver; //自定义名称上下一致

index index.html index.htm;

........

}

分配服务器策略

轮询：按照时间的顺序进行分配

weight：代表权重 默认1，权重越高分配的客户端越多

ip\_hash：每个请求按访问IP的hash结果分配的，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

fair：按照后端服务器响应时间来分配

动静分离配置