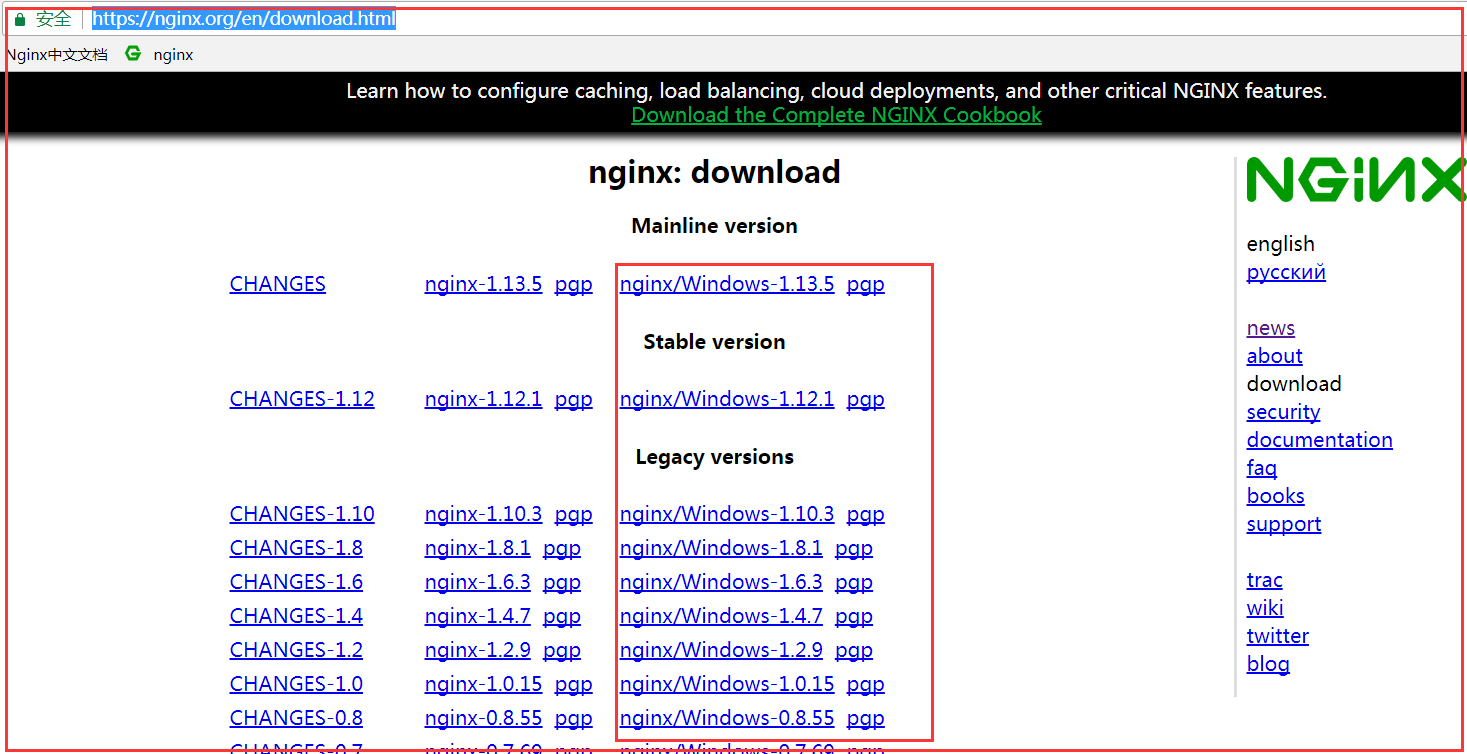
# **Nginx安装**

### **1. windows安装**

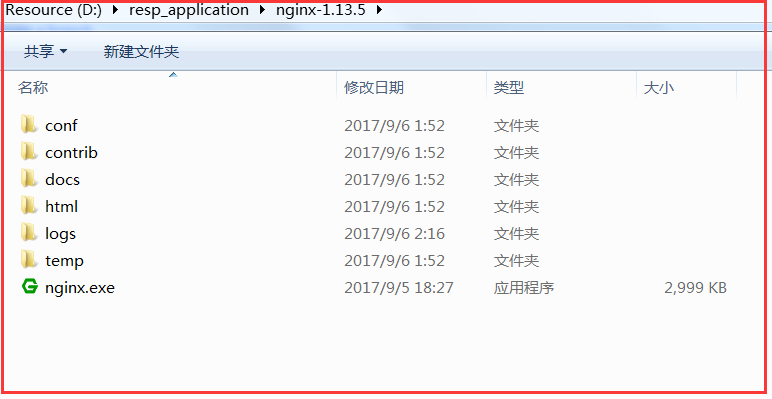
官方网站下载地址：

https://nginx.org/en/download.html

如下图所示，下载对应的版本的nginx压缩包，解压到自己电脑上存放软件的文件夹中即可



解压完成后，文件目录结构如下：



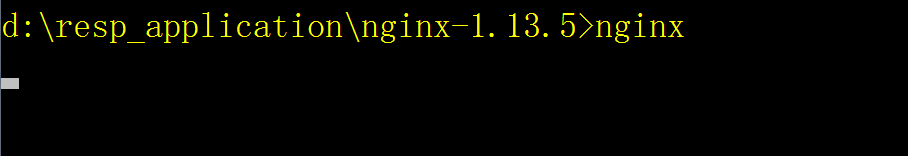
### 

### **启动nginx**

1） 直接双击该目录下的nginx.exe，即可启动nginx服务器

2） 命令行计入该文件夹，执行nginx命令，也会直接启动nginx服务器

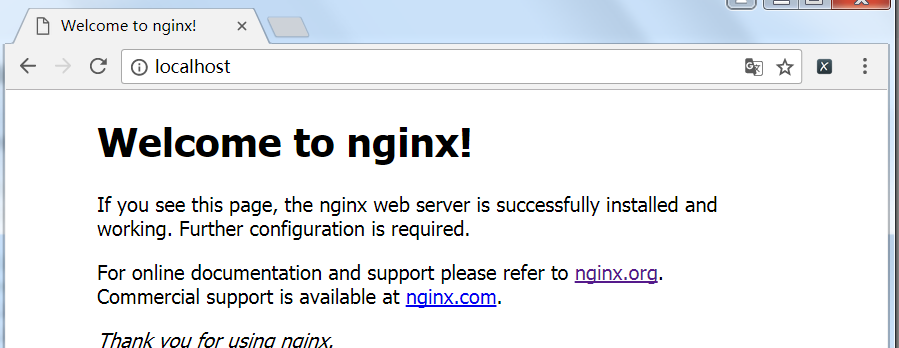
D:/resp\_application/nginx-1.13.5> nginx



### **访问nginx**

打开浏览器，输入地址：http://localhost，访问页面，出现如下页面表示访问成功

### 



### **停止nginx**

命令行进入nginx根目录，执行如下命令，停止服务器：

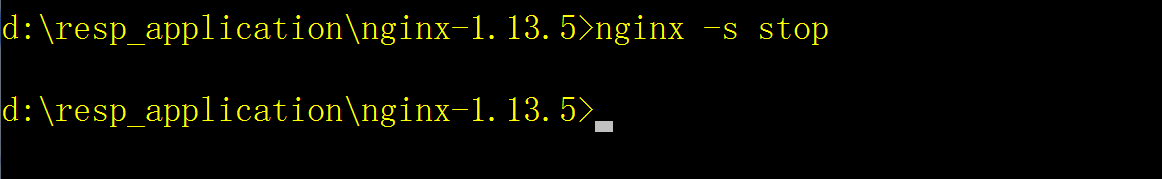
# 强制停止nginx服务器，如果有未处理的数据，丢弃

D:/resp\_application/nginx-1.13.5> nginx -s stop

# 优雅的停止nginx服务器，如果有未处理的数据，等待处理完成之后停止

D:/resp\_application/nginx-1.13.5> nginx -s quit

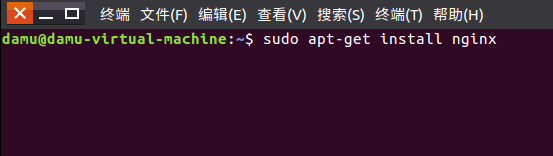
### 



### **2. ubuntu安装**

按照正常软件的安装方式，直接通过如下命令进行安装：

$ sudo apt-get install nginx



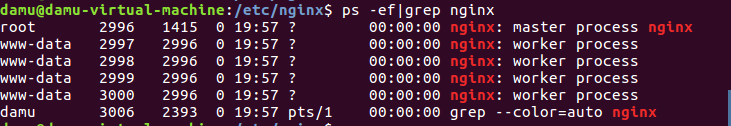
安装完成即可，在/usr/sbin/目录下是nginx命令所在目录，在/etc/nginx/目录下是nginx所有的配置文件，用于配置nginx服务器以及负载均衡等信息

##### 

##### **查看nginx进程是否启动**

$ ps -ef|grep nginx

nginx会自动根据当前主机的CPU的内核数目创建对应的进程数量(当前ubuntu主机是2核4线程配置)



备注：这里启动的服务进程其实是4个进程，因为nginx进程在启动的时候，会附带一个守护进程，用于保护正式进程不被异常终止；如果守护进程一旦返现nginx继承被终止了，会自动重启该进程。

守护进程一般会称为master进程，业务进程被称为worker进程

##### 

##### **启动nginx服务器命令**

直接执行nginx会按照默认的配置文件进行服务器的启动

$ nginx

IMG_263

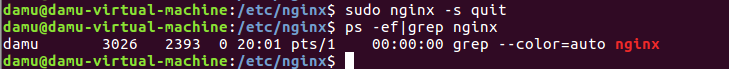
##### 

##### **停止nginx服务命令**

和windows系统执行过程一样，两种停止方式

$ nginx -s stopor

$ nginx -s quit



##### 

##### **重新启动加载**

同样也可以使用命令reopen和reload来重新启动nginx或者重新加载配合着文件。

# **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

main # 全局配置

events { # nginx工作模式配置

}

http { # http设置 ....

server { # 服务器主机配置 ....

location { # 路由配置 ....

}

location path {

....

}

location otherpath {

....

}

}

server {

....

location {

....

}

}

upstream name { # 负载均衡配置 ....

}

}

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

### **event 模块**

上干货

event {

worker\_connections 1024;

multi\_accept on;

use epoll;

}

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！

http {

##

# 基础配置

##

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

keepalive\_timeout 65;

types\_hash\_max\_size 2048;

# server\_tokens off;

# server\_names\_hash\_bucket\_size 64;

# server\_name\_in\_redirect off;

include /etc/nginx/mime.types;

default\_type application/octet-stream;

##

# SSL证书配置

##

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2; # Dropping SSLv3, ref: POODLE ssl\_prefer\_server\_ciphers on;

##

# 日志配置

##

access\_log /var/log/nginx/access.log;

error\_log /var/log/nginx/error.log;

##

# Gzip 压缩配置

##

gzip on;

gzip\_disable "msie6";

# gzip\_vary on;

# gzip\_proxied any;

# gzip\_comp\_level 6;

# gzip\_buffers 16 8k;

# gzip\_http\_version 1.1;

# gzip\_types text/plain text/css application/json application/javascript

text/xml application/xml application/xml+rss text/javascript;

##

# 虚拟主机配置

##

include /etc/nginx/conf.d/\*.conf;

include /etc/nginx/sites-enabled/\*;

1) 基础配置

sendfile on：配置on让sendfile发挥作用，将文件的回写过程交给数据缓冲去去完成，而不是放在应用中完成，这样的话在性能提升有有好处

tc\_nopush on：让nginx在一个数据包中发送所有的头文件，而不是一个一个单独发

tcp\_nodelay on：让nginx不要缓存数据，而是一段一段发送，如果数据的传输有实时性的要求的话可以配置它，发送完一小段数据就立刻能得到返回值，但是不要滥用哦

keepalive\_timeout 10：给客户端分配连接超时时间，服务器会在这个时间过后关闭连接。一般设置时间较短，可以让nginx工作持续性更好

client\_header\_timeout 10：设置请求头的超时时间

client\_body\_timeout 10:设置请求体的超时时间

send\_timeout 10：指定客户端响应超时时间，如果客户端两次操作间隔超过这个时间，服务器就会关闭这个链接

limit\_conn\_zone $binary\_remote\_addr zone=addr:5m ：设置用于保存各种key的共享内存的参数，

limit\_conn addr 100: 给定的key设置最大连接数

server\_tokens：虽然不会让nginx执行速度更快，但是可以在错误页面关闭nginx版本提示，对于网站安全性的提升有好处哦

include /etc/nginx/mime.types：指定在当前文件中包含另一个文件的指令

default\_type application/octet-stream：指定默认处理的文件类型可以是二进制

type\_hash\_max\_size 2048：混淆数据，影响三列冲突率，值越大消耗内存越多，散列key冲突率会降低，检索速度更快；值越小key，占用内存较少，冲突率越高，检索速度变慢

2) 日志配置

access\_log logs/access.log：设置存储访问记录的日志

error\_log logs/error.log：设置存储记录错误发生的日志

3) SSL证书加密

ssl\_protocols：指令用于启动特定的加密协议，nginx在1.1.13和1.0.12版本后默认是ssl\_protocols SSLv3 TLSv1 TLSv1.1 TLSv1.2，TLSv1.1与TLSv1.2要确保OpenSSL >= 1.0.1 ，SSLv3 现在还有很多地方在用但有不少被攻击的漏洞。

ssl prefer server ciphers：设置协商加密算法时，优先使用我们服务端的加密套件，而不是客户端浏览器的加密套件

4) 压缩配置

gzip 是告诉nginx采用gzip压缩的形式发送数据。这将会减少我们发送的数据量。

gzip\_disable 为指定的客户端禁用gzip功能。我们设置成IE6或者更低版本以使我们的方案能够广泛兼容。

gzip\_static 告诉nginx在压缩资源之前，先查找是否有预先gzip处理过的资源。这要求你预先压缩你的文件（在这个例子中被注释掉了），从而允许你使用最高压缩比，这样nginx就不用再压缩这些文件了（想要更详尽的gzip\_static的信息，请点击这里）。

gzip\_proxied 允许或者禁止压缩基于请求和响应的响应流。我们设置为any，意味着将会压缩所有的请求。

gzip\_min\_length 设置对数据启用压缩的最少字节数。如果一个请求小于1000字节，我们最好不要压缩它，因为压缩这些小的数据会降低处理此请求的所有进程的速度。

gzip\_comp\_level 设置数据的压缩等级。这个等级可以是1-9之间的任意数值，9是最慢但是压缩比最大的。我们设置为4，这是一个比较折中的设置。

gzip\_type 设置需要压缩的数据格式。上面例子中已经有一些了，你也可以再添加更多的格式。

5) 文件缓存配置

open\_file\_cache 打开缓存的同时也指定了缓存最大数目，以及缓存的时间。我们可以设置一个相对高的最大时间，这样我们可以在它们不活动超过20秒后清除掉。

open\_file\_cache\_valid 在open\_file\_cache中指定检测正确信息的间隔时间。

open\_file\_cache\_min\_uses 定义了open\_file\_cache中指令参数不活动时间期间里最小的文件数。

open\_file\_cache\_errors 指定了当搜索一个文件时是否缓存错误信息，也包括再次给配置中添加文件。我们也包括了服务器模块，这些是在不同文件中定义的。如果你的服务器模块不在这些位置，你就得修改这一行来指定正确的位置。

### **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

server {

listen 80;

server\_name localhost 192.168.1.100;

root /nginx/www;

index index.php index.html index.html;

charset utf-8;

access\_log logs/access.log;

error\_log logs/error.log;

......

}

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 / {

root /nginx/www;

index index.php index.html index.htm;

}

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

location / {

proxy\_pass http://localhost:8888;

proxy\_set\_header X-real-ip $remote\_addr;

proxy\_set\_header Host $http\_host;

}

uwsgi配置

wsgi模式下的服务器配置访问方式

location / {

include uwsgi\_params;

uwsgi\_pass localhost:8888

}

### **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

upstream name {

ip\_hash;

server 192.168.1.100:8000;

server 192.168.1.100:8001 down;

server 192.168.1.100:8002 max\_fails=3;

server 192.168.1.100:8003 fail\_timeout=20s;

server 192.168.1.100:8004 max\_fails=3 fail\_timeout=20s;

}

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