搭建laravel环境

## **安装 Laravel**

### **Laravel 简介**

[Laravel](https://laravel.com/" \o "null" \t "https://www.cnblogs.com/jikexianfeng/p/_blank) 是一套简洁、优雅的 PHP Web 开发框架。它可以让你从面条一样杂乱的代码中解脱出来；它可以帮你构建一个完美的 web APP，而且每行代码都可以简洁、富于表达力。

### **安装依赖**

由于默认的 **yum** 源 php 版本低于 Laravel 要求，所以需要添加第三方源：

rpm -Uvh https://mirror.webtatic.com/yum/el7/epel-release.rpm

rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm

安装 **nginx**、**php** 及相关依赖：

yum -y install nginx mariadb-server php70w php70w-fpm php70w-mysql php70w-mcrypt php70w-dom php70w-mbstring

### **配置 Mariadb（MySQL）**

使用以下命令启动 **mysql** 并设为开启启动：

systemctl start mariadb

systemctl enable mariadb

首次启用 **mysql** 时，我们需要执行以下指令进行配置：

mysql\_secure\_installation

过程中除下图中两次 输入及确认密码 外，一路按回车键选择**默认值**即可。

IMG_256

### **安装 Composer**

[Composer](https://getcomposer.org/" \o "null" \t "https://www.cnblogs.com/jikexianfeng/p/_blank) 是 **php** 的依赖管理工具，我们将使用它下载 Laravel 安装包。

#### **下载 Composer**

curl -sS https://getcomposer.org/installer | php

mv composer.phar /usr/local/bin/composer

设置环境变量，只有这样安装后系统才能找到 laravel 的执行文件：

export PATH=$PATH:/root/.config/composer/vendor/bin

#### **设置 Swap**

为了避免 composer 安装应用过程中出现内存不足的问题，我们预先设置一下 **swap** ：(Swap 分区在系统的物理内存不够用的时候，把硬盘空间中的一部分空间释放出来，以供当前运行的程序使用)

/bin/dd if=/dev/zero of=/var/swap.1 bs=1M count=1024

/sbin/mkswap /var/swap.1

/sbin/swapon /var/swap.1

### **Laravel 安装**

使用 Composer 安装 Laravel：

composer global require "laravel/installer"

## **使用 Laravel**

### **新建项目**

进入 **/home** 目录，我们创建一个 laravel 项目，命名为 **blog**：

cd /home

laravel new blog

### **更新项目依赖**

进入 **blog** 项目，使用 composer update --no-scripts 更新项目依赖：

cd blog

composer update --no-scripts

### **配置项目**

#### **配置目录权限**

为了运行 Laravel，我们需要为一些项目目录配置权限：

sudo chmod 775 /home/blog/storagesudo chmod 775 /home/blog/bootstrap/cache

#### **生成密钥**

查看 blog 目录下是否包含 **.env** 文件，如果不存在，则右击 .env.example 文件，将其重命名为 **.env**。

我们使用以下命令来生成一串密钥：

php artisan key:generate

执行后会得到如下输出：

Application key [...] set successfully.

点击打开 /config/app.php，找到如下一行：

'key' => env('APP\_KEY'),

将生成的**密钥**填入（中括号中部分）：

'key' => env('APP\_KEY', '...'),

### **测试启动**

在 **blog** 目录下，我们使用下面命令来启用 开发服务器 测试访问：

sudo php artisan serve --host=0.0.0.0 --port=80

接着，我们可以打开http：//IP 测试访问。

## **配置 nginx**

上面的 **serve** Artisan 命令一般只用于本地开发，而生产环境中我们需要使用 Web 服务器，这里我们选用了 **nginx**。

### **启动 php-fpm**

首先我们先按 Ctrl + C 停止掉刚刚的 **serve**。

在 nginx 中，我们通过 **php-fpm** 来调用 php，通过下面命令启动 **php-fpm**：

systemctl start php-fpm

systemctl enable php-fpm

可以使用下面的命令查看 **php-fpm** 是否启动 ： (php-fpm 默认监听 9000 端口)

netstat -nlpt | grep php-fpm

### **编辑配置**

打开 /etc/nginx/nginx.conf，备注或移除如下内容：

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name \_;

root /usr/share/nginx/html;

...

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

}

}

在 /etc/nginx/conf.d 目录下 创建 php.conf，然后在该文件中添加如下内容

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name \_;

root /home/blog/public;

index index.php index.html index.htm;

location / {

try\_files $uri $uri/ /index.php?$query\_string;

}

location ~ .php$ {

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include fastcgi\_params;

}

}

启动 Nginx：

systemctl start nginx

systemctl enable nginx

我们可以打开http：//IP 测试访问。