**一文教您如何通过Docke搭建反向代理 Ngnix，并配置Https SSL证书**

**一、背景**

小哈最近收到阿里云短信，提示个站 [www.exception.site](https://juejin.cn/post/www.exception.site) 的云盾 SSL 证书(Https 证书)即将到期，需要赶快续费，不然无法继续使用Https协议来访问网站！

这个SSL证书当时用的是阿里云免费型的，有效期为1年，到期后, 如果想继续使用这个证书，就得续费，而且费用还不低！



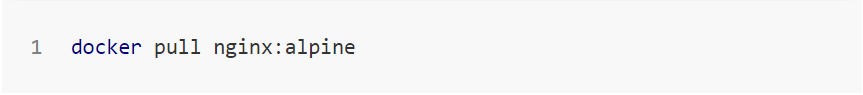
其实，要想继续使用Https协议，我们可以在阿里云上再次申请一张 SSL 证书就可以了，时间一年，嗨呀，又能免费用一年。

本文小哈就主要介绍如何在Ngnix上配置SSL证书，从而让我们的网站能够使用Https 来访问，另外再附带说一下如何配置反向代理。

**二、Docker 快速安装&搭建 Ngnix 环境**

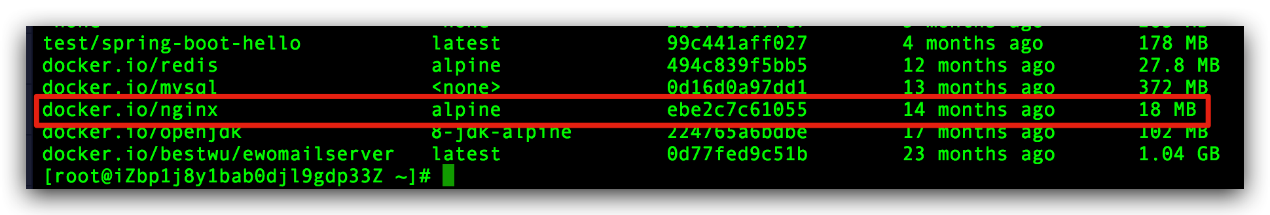
小哈的个站Ngnix搭建在容器当中，也就顺便说一下如何通过 Docker 快速安装&搭建 Ngnix环境。

**2.1 下载 Nginx 镜像**

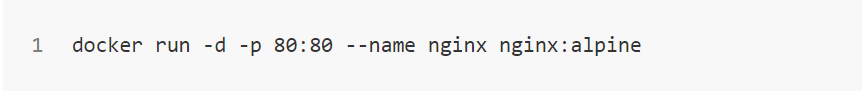


PS：我这里用的是alipne轻量级的镜像.

下载完成后，通过 docker images 命令检查一下镜像是否下载成功：

检查ngnix镜像是否下载成功

**2.2 先以简单的方式运行镜像**



* **-p 80:80:** 将容器的 80 端口映射到宿主机的 80 端口上；
* **-d:** 以后台方式运行镜像；
* **--name:** 指定容器的名称为 nginx;

命令执行完成后，通过**docker ps**命令确认一下容器是否启动成功。确认成功后，再访问一下 **80** 端口，看看nginx服务是否启动成功：

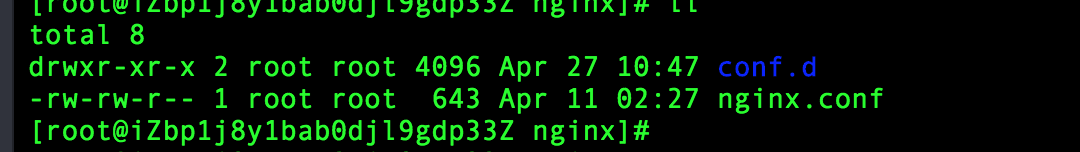
**nginx访问成功**

**2.3 复制相关配置文件至宿主机**

复制运行中nginx相关配置文件到宿主机的指定路径下：



复制完成后，查看指定路径的配置文件，如下：

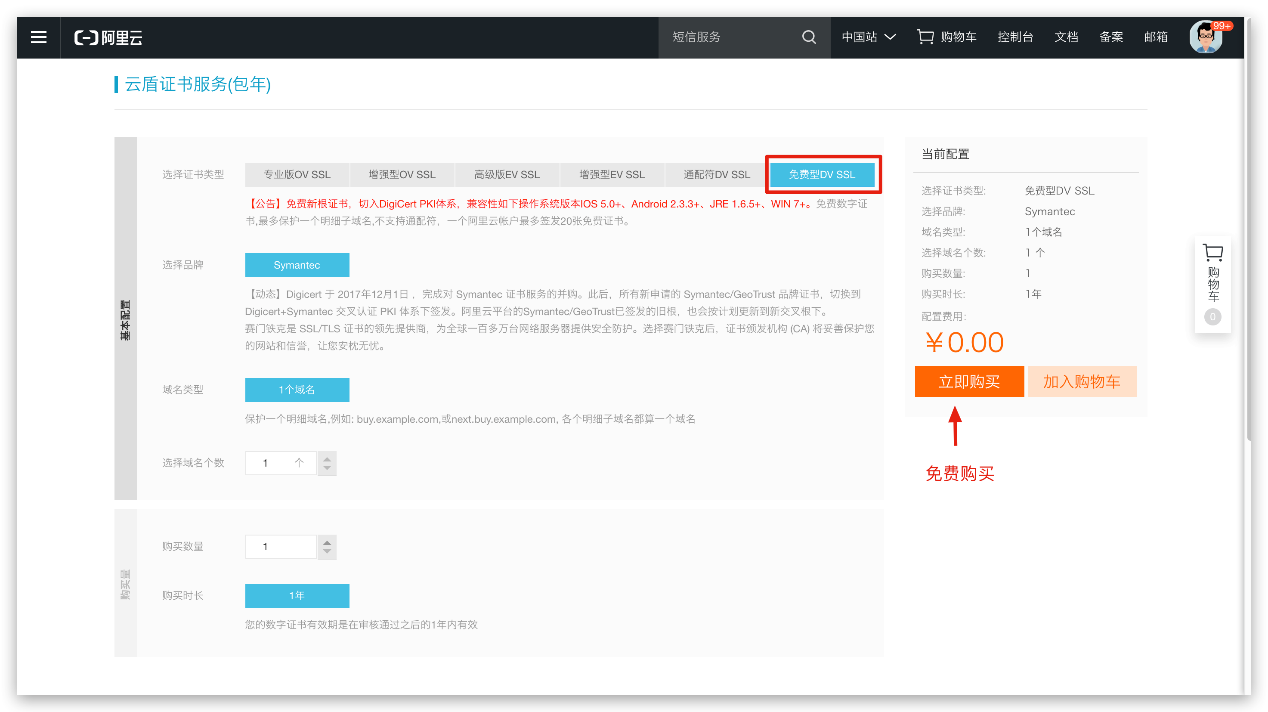


**三、配置 SSL**

**3.1 免费获取 SSL 证书**

访问链接 [common-buy.aliyun.com/?spm=5176.2…](https://link.juejin.cn/?target=https%3A%2F%2Fcommon-buy.aliyun.com%2F%3Fspm%3D5176.2020520163.cas.3.4bf91BFD1BFDcm%26commodityCode%3Dcas%23%2Fbuy)：

**PS: 注意，未注册阿里云账号的童鞋，需要先注册一个账号哟~**



选择**免费 DV SSL 证书** -> **立即购买**，完成后，进入阿里云后台查看：



这里需要申请签发，填写相关信息后，等待签发通过即可。



签发通过后，我们下载证书：

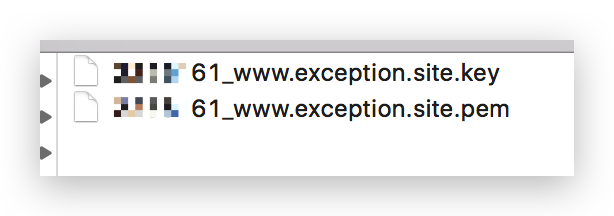




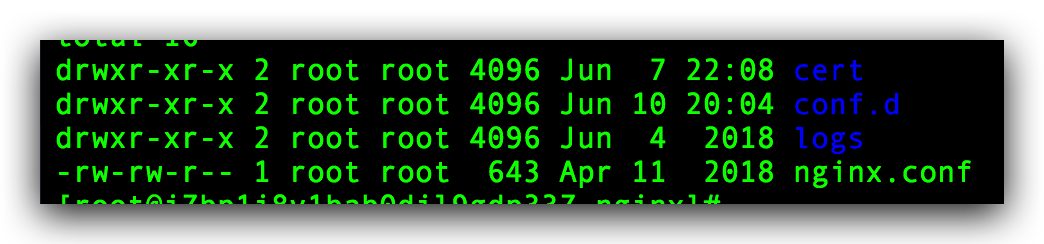
这里我选择的是Ngnix,当然了，你也可以选择Tomcat、Apache、IIS等。

**3.2 解压上传到服务器**

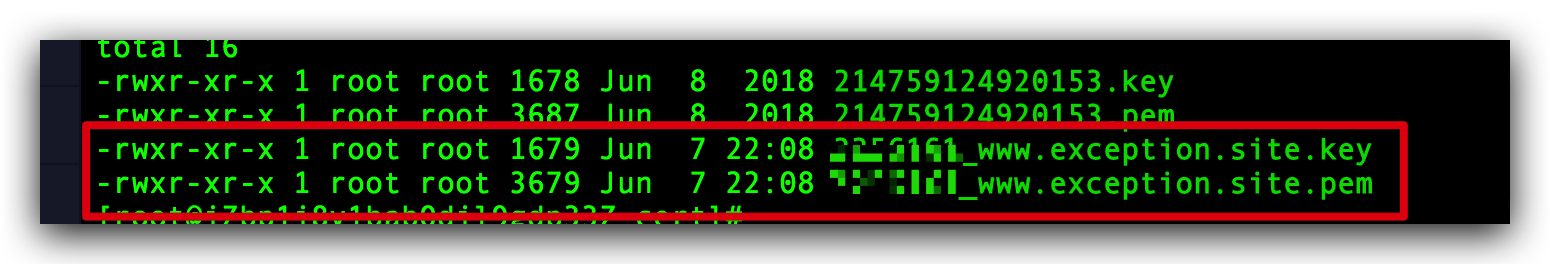
下载下来后，解压，我们可以看到包含两个文件：



接下来，我们先在容器的挂载目录下创建**cert**证书目录：



将解压后的两个文件上传至该目录中。



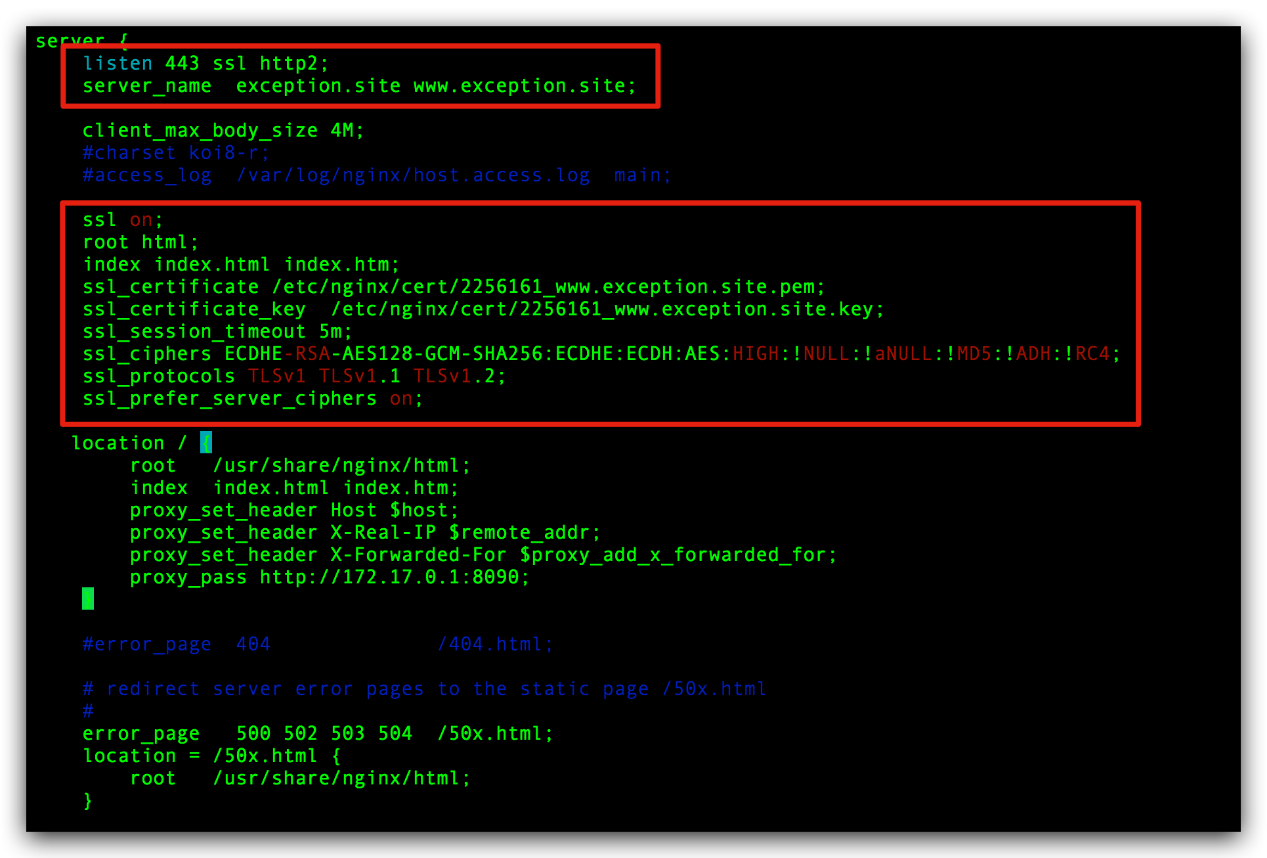
**3.3 配置 Ngnix SSL**

这里需要说一下，小哈运行在 docker 容器中的 nginx 版本，信息如下：



因为我的 nginx 相对版本较高，可能和大家的配置文件目录有些不同，但整体的没啥大的问题，该怎么配置，还是怎么配置：

首先创建 **ssl.conf** 配置文件，专门用来配置 **ssl**相关参数：

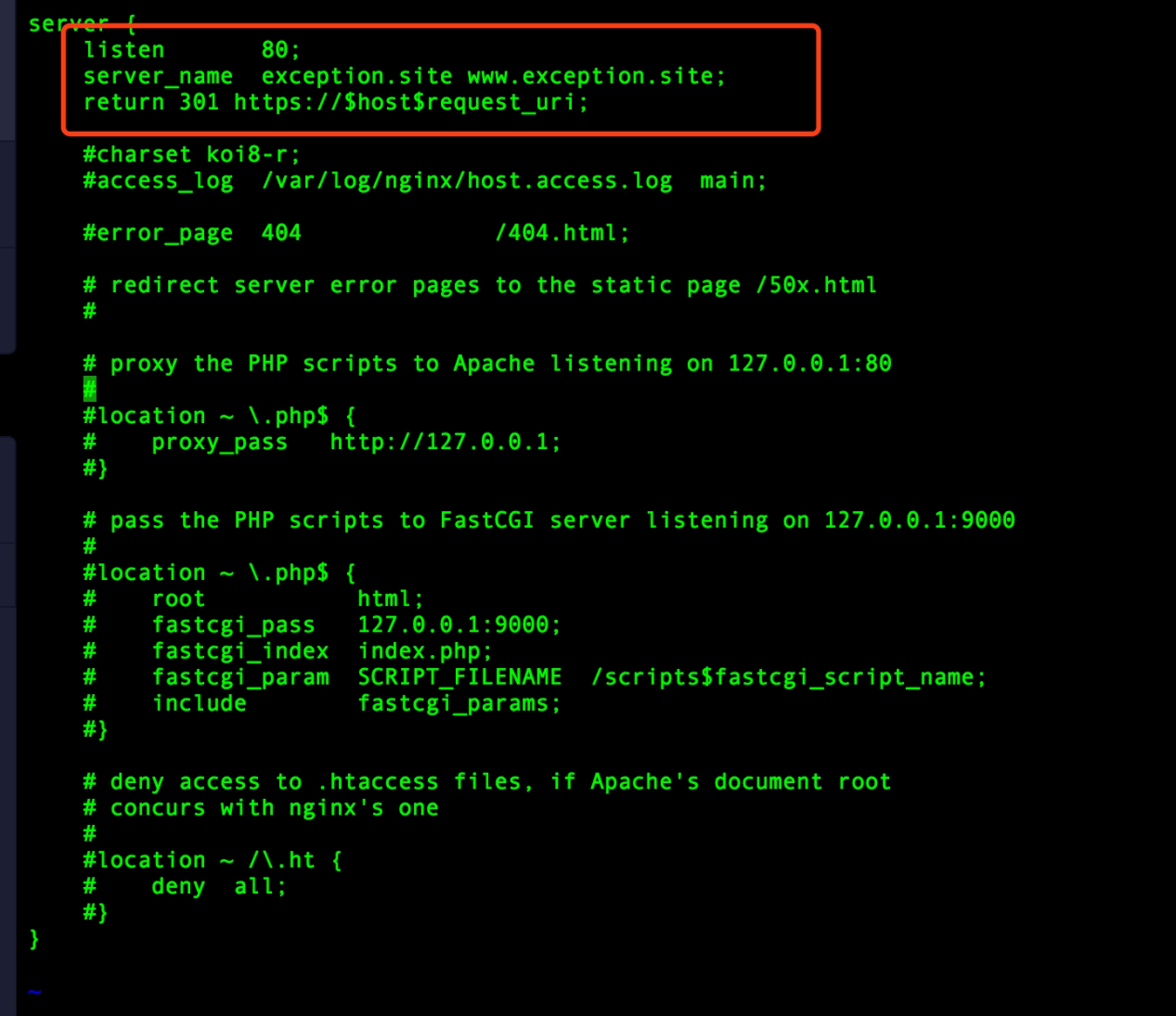






根据阿里云的文档说明加上 **ssl**相关参数，如**ssl**证书的两个文件目录，另外同时监听 **443** 端口。

再贴下 **defaut.conf**配置文件内容：

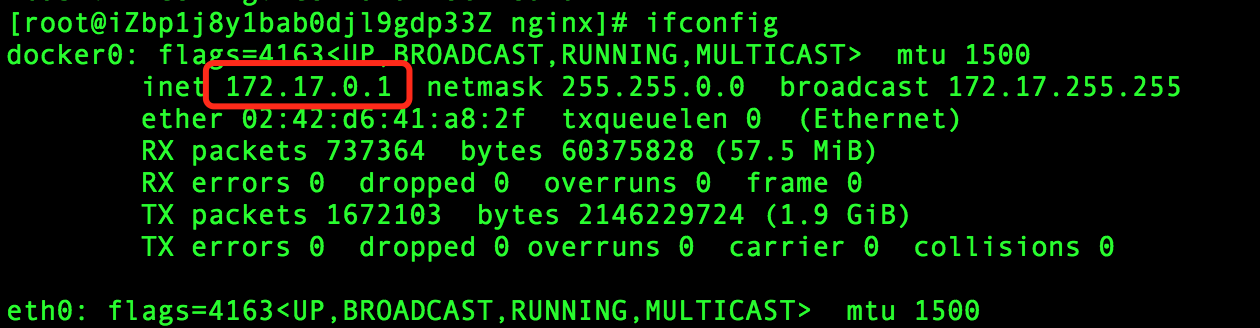




可以看到，主要对 **http 80**端口主要做了一下 **301**重定向，也就是说，如果前端使用的 **http** 访问，这里会重定向到**https 协议**。

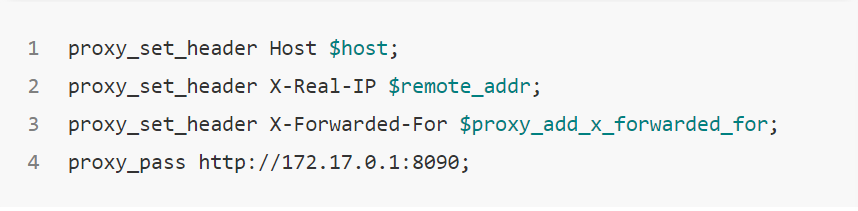
**四、配置反向代理**

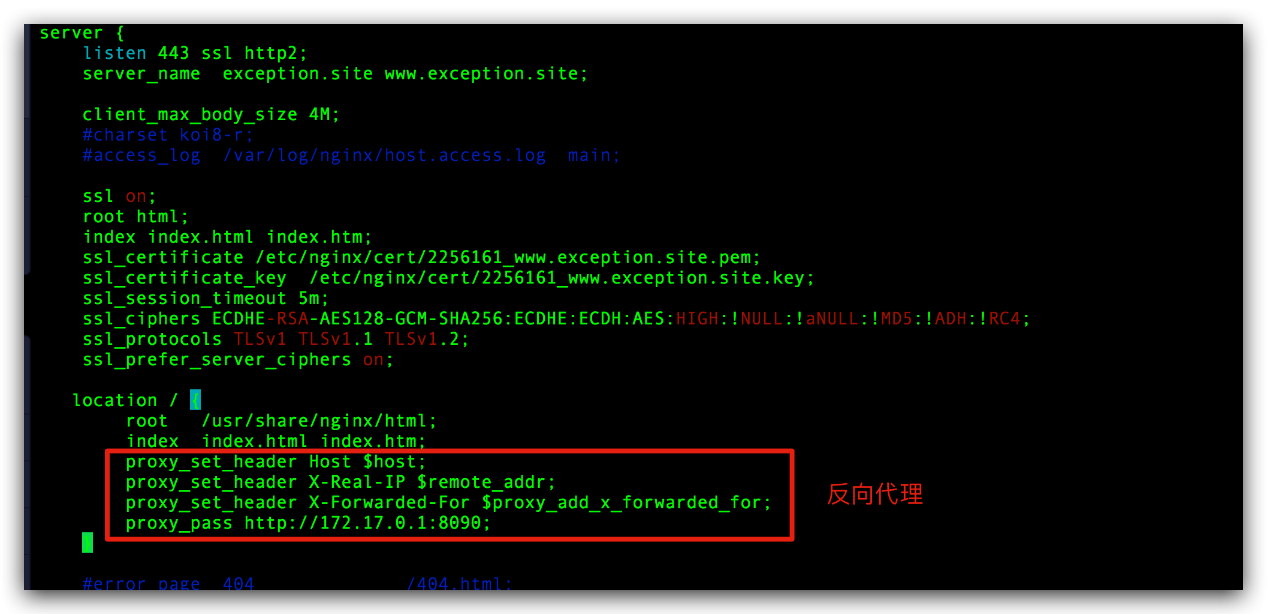
再来说说如何配置反向代理，先通过命令 **ifconfig** 命令获取 **docker0**网卡的 **ip** 地址：



PS: 因为我需要反向代理的服务部署在本机的 Docker 容器中，所以用的这个 IP，如果你用的外网其他地址，可自行修改。

由于上面我们已经新建了一个 **ssl.conf** 配置文件，这里就直接将反向代理的配置写在里面：

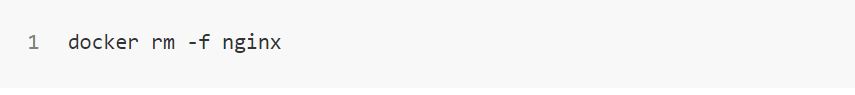




通过配置，我们反向代理到了 **8090** 端口的 **Spring Boot** 服务。

**五、重启 Nginx 容器**

先删除之前启动的 nginx 容器：



成功后，回到配置文件的主目录，执行：

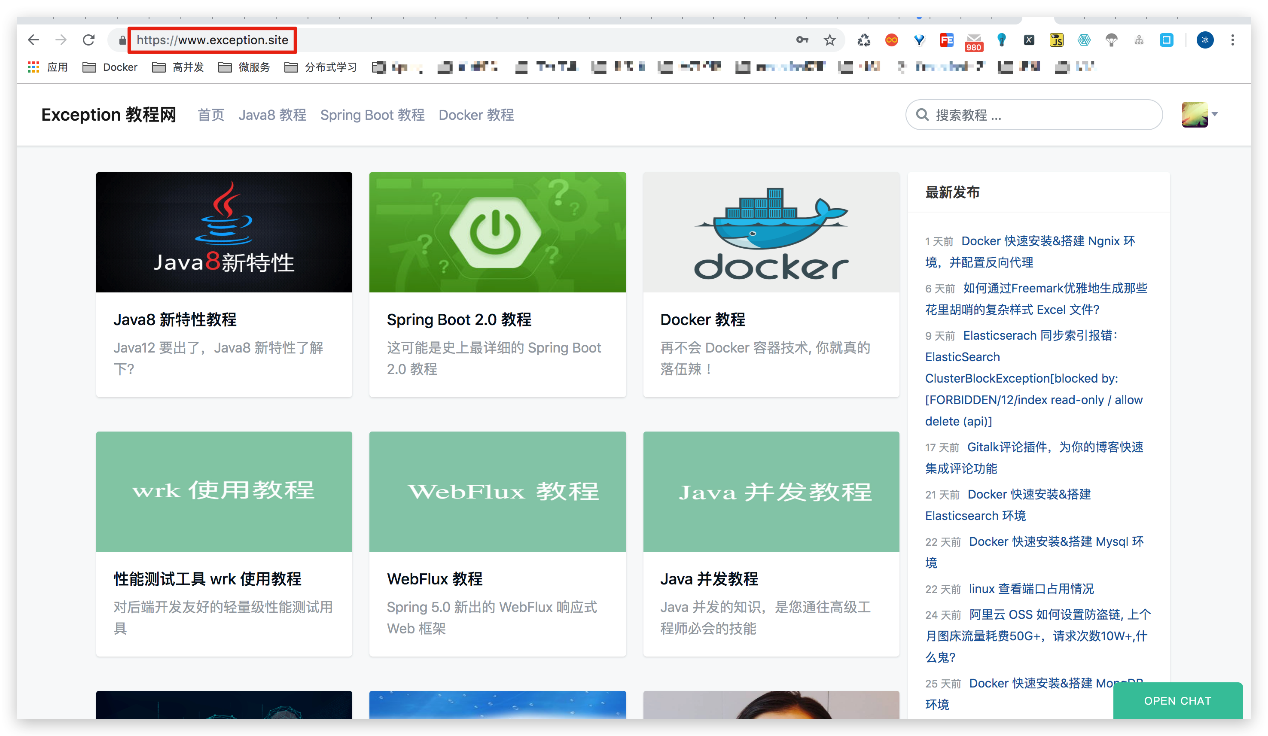


参数说明：

* **-p 80:80 -p 443:443**：将容器的 80、443 端口映射到主机的 80、443 端口；
* **-v/docker/nginx/nginx.conf:/etc/nginx/nginx.conf**：将容器中的/etc/nginx/nginx.conf 文件挂载到宿主机中的 /docker/nginx/nginx.conf 文件；
* **-v /docker/nginx/conf.d:/etc/nginx/conf.d**：将容器中 /etc/nginx/conf.d 目录挂载到宿主机中的 conf.d 目录下；
* **-v /docker/nginx/logs:/var/log/nginx**：将容器中的 /var/log/nginx 目录挂载到宿主机中的 /docker/nginx/logs 目录下, 用以查看 Nginx 日志；
* **-v /docker/nginx/cert:/etc/nginx/cert**: 将容器中的 /etc/nginx/cert 证书目录挂载到宿主机中的 /docker/nginx/cert 证书目录下；

注意：因为我使用的阿里云服务器，同时还需要将 443 端口加入到安全组中，才能保证外界的正常访问 https 端口

运行命令成功后，通过 https 协议来访问我们的网站域名 [www.exception.site](https://juejin.cn/post/www.exception.site)，看看是否能够正常访问，并且方向代理到 8090 端口的 Spring Boot 服务上：



OK! 大工告成。

**六 、最后**

本文中，我们学习了如何通过**Docker**快速搭建 **Nginx** 环境，并对其配置 **SSL** 证书和反向代理，让网站能够以**Https**协议来访问。希望童鞋们看完本文能够有所收获，下期见~

**转载自稀土掘金作者犬小哈，欢迎关注作者微信公众号: 小哈学Java**