如何快速部署上线项目

==========

大家好呀，我是苍何。

今天在群里面看到有小伙伴反馈说，面试的时候一被问到简历中的项目还没上线，就不继续问了，感觉挺奇葩的，要知道就校招来说，项目本身大部分都是练手的项目，上线也得花费很多的成本啊。

![群友反馈项目没上线就没下文](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/4bd02dad01654202b0233f32f64cbab0~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=1636&h=1254&s=198357&e=png&a=1&b=f0f0f0 "群友反馈项目没上线就没下文")

就 Java 应用来说本身是很吃内存的，再者现在不拿个微服务项目挂简历又拿不出手，但一上线微服务，服务器资源耗费将会是之前的好几倍。就拿 PmHub 来说，之前单体的时候，1 核 2G 的服务器，跑的还可以。

但一换上微服务，即使升级为 4 核 8G， 服务还没全开的情况下，内存一样被打满，不得已把运行 CI/CD 的 Jenkins 给关了才得以「保命」。

![我的服务器资源占用整体情况](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/6b559eb20ae04a938b769b9048f2d33f~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=1916&h=261&s=82033&e=png&a=1&b=2b2b2b "我的服务器资源占用整体情况")

![我的服务器资源占用细节](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/d12cf9fcea8b4fa894d91ffe5bdfe3e6~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=1683&h=812&s=194804&e=png&a=1&b=262626 "我的服务器资源占用细节")

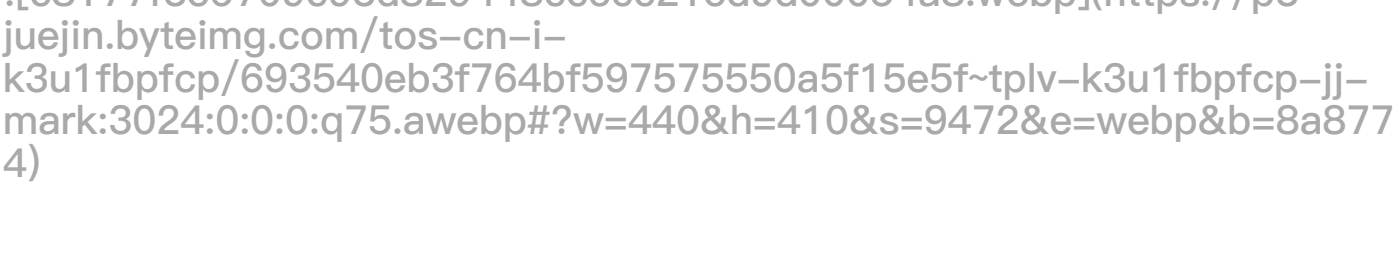
而且该说不说，现在的云服务器续费也是贵的离谱，省下好几个月的余粮才咬牙升级配置。（为了项目有个舒服的体验环境也是拼了）

![含泪升级云服务器](https://p3-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/484fe9b092cb4bf799606b8aef2bb47e~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=946&h=574&s=35204&e=jpg&b=1974fd

”含泪升级云服务器”)

说回面试需要项目上线的问题，我觉得主要有两个办法，一是本地电脑部署外挂内网穿透出去，这样也省去买域名和服务器一堆操作，是性价比最高的。但缺点就是**电脑不能关**，直到拿到 offer 结束。

当然条件允许的话也可以体验一把部署上线的**快感**。为什么说是快感呢？有经验的开发应该清楚，每次上线都是激动人心的时刻，线下爱你千万遍，不如线上见一面。上线一次就顺利的算是运气好，很多情况，上线总要出问题。



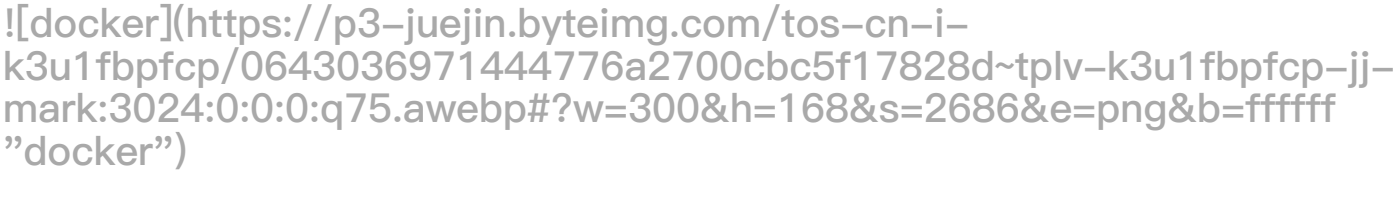
今天分享一下我们项目的容器化部署实战，希望对你有帮助。

什么是 Docker 容器
==========

当把环境从一个服务器迁移到另一个服务器，当你新买了电脑，会遇上要重新安装一系列的软件，比如 JDK、Tomcat、Maven、Redis 等，而安装完之后每一次都需要再去配置环境，但是这些都是重复性的工作，有什么办法可以解决呢?

玩过云服务器的听说过有镜像一说，将所有的软件和需要的配置都打包好做成镜像，下次需要的时候只需要安装镜像就可以。

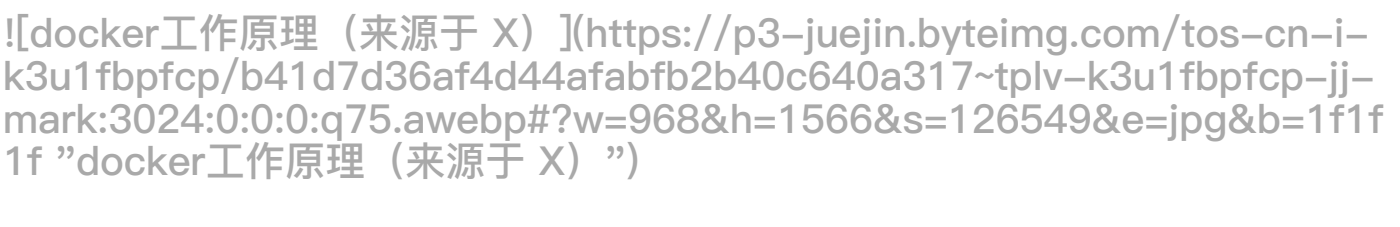
类似的，Docker 就是解决这种问题的，将所有的软件程序都放在 Docker 仓库中，我们只需要在电脑上下载 Docker 即可，不需要再去进行繁琐的安装和配置了，只需要几个命令就可以将我们的环境搭建好，这就是 Docker。



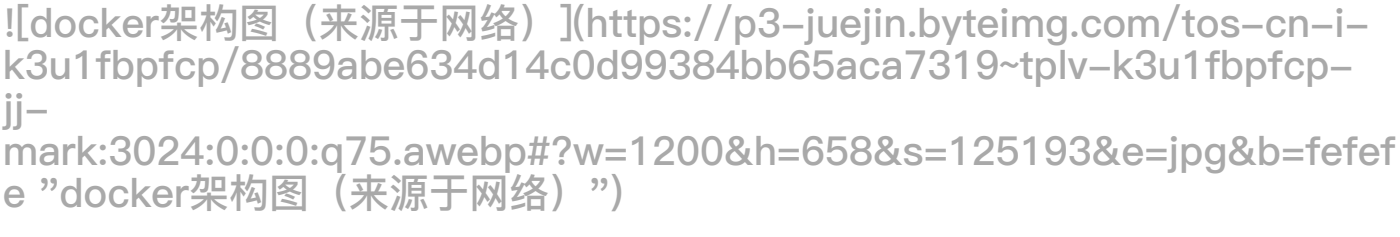
总结来说， Docker 是一个开源的应用容器引擎；是一个轻量级容器技术，Docker 支持将软件编译成一个镜像；然后在镜像中各种软件做好配置，将镜像发布出去，其他使用者可以直接使用这个镜像；运行中的这个镜像称为容器，容器启动是非常快速的。

Docker 架构

Docker 的工作原理如下：



通过下图可以得知，Docker在运行时分为Docker引擎（服务端守护进程）和客户端工具，我们日常使用各种docker命令，其实就是在使用客户端工具与Docker引擎进行交互。



Dockerfile

在 Docker 中构建镜像最常用的方式，就是使用Dockerfile。Dockerfile是一个用来构建镜像的文本文件，文本内容包含了一条条构建镜像所需的指令和说明。官方文档：[docs.docker.com/engine/refe...](http://cxyroad.com/"https://docs.docker.com/engine/reference/builder%E3%80%82")

就是可以存放构建启动指令，可以用来动态构建 docker 镜像容器的文件。

为什么要用 Docker-Compose

因为是微服务项目，经常来说，单独去启动每个服务太麻烦了，而且有些服务间有启动顺序，有需要数据库服务容器、缓存容器等这些。

这就需要用到 Docker-Compose，它是用于定义和运行多容器 Docker 应用程序的工具。通过Compose，您可以使用YAML文件来配置应用程序所需要的服务。然后使用一个命令，就可以通过YAML配置文件创建并启动所有服务。

我们的项目用的也是 Docker-Compose，可以做到一条命令就可以启动所有服务容器，主打一个**高效**。

项目上线实战
======

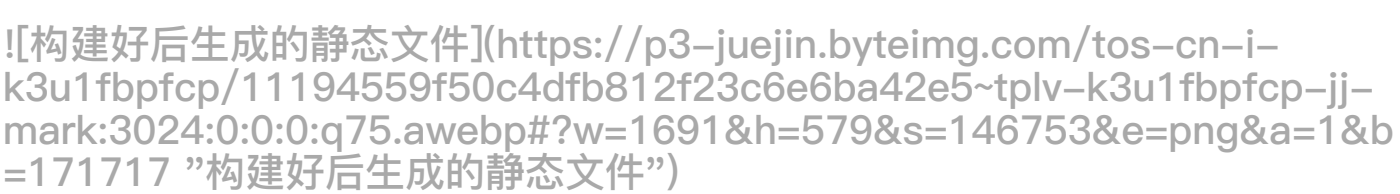
本地构建

1、前端构建

前端的 vue 进行构建的时候还是比较吃资源的，如果资源配置不足的情况下，可以在本地电脑构建，然后将构建好后的 dist 文件上传到服务器即可。

> 如果.gitignore 忽略了 dist 的提交，记得注释掉！

```
npm run build:prod
```

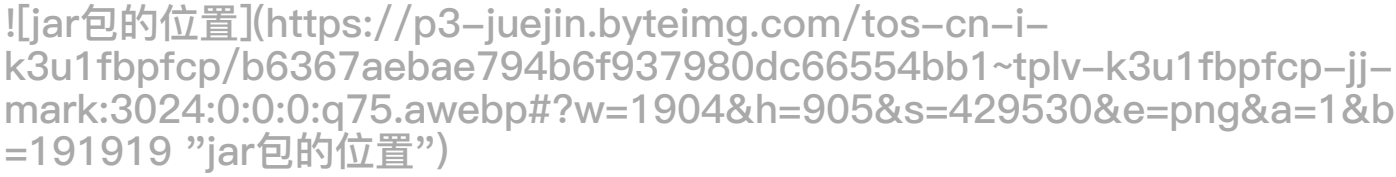


后端构建

后端打包构建完全可以放在服务器上，后面配合 CI/CD 构建脚本，可以做到自动化打包构建部署上线，它其实也就是一条命令即可。

```
# 切换到目录
cd /你的安装目录/
# 包构建项目
mvn -T 1C clean package -Dmaven.test.skip=true -Dmaven.compile.fork=true
```

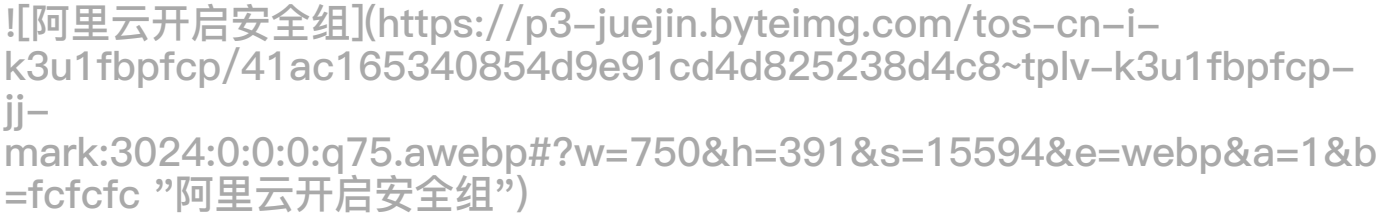
构建完成后，在每个项目的 target 下面就可以看到打包好的 jar 包：



阿里云开启安全组

如果服务器是阿里云的话，在操作云服务器之前，先需要开启阿里云的安全组，进行端口的放开。一般来说，云服务器的端口由以下两个控制。

* 安全组放开端口
* 防火墙开启端口



下载项目

接下来就是在云服务器上 clone 项目：

```
# 进去目录
cd /你的安装目录

git clone git@github.com:laigeoffer/pmhub.git

```

拷贝 jar 包

微服务的 jar 包分散在各自的模块，不大方便操作，因此提供了一键复制 jar 包合相关配置文件到统一地方的 shell 脚本如下：

```
#!/bin/sh

# 复制项目的文件到对应docker路径，便于一键生成镜像。
usage() {
echo "Usage: sh copy.sh"
exit 1
}


# copy sql
echo "begin copy sql "
cp ../sql/pmhub_20240305.sql ./mysql/db
cp ../sql/pmhub_nacos_20240423.sql ./mysql/db

# copy html
echo "begin copy html "
cp -r ../pmhub-ui/dist/** ./nginx/html/dist


# copy jar
echo "begin copy pmhub-gateway "
cp ../pmhub-gateway/target/pmhub-gateway.jar ./pmhub/gateway/jar

echo "begin copy pmhub-auth "
cp ../pmhub-auth/target/pmhub-auth.jar ./pmhub/auth/jar

echo "begin copy pmhub-monitor "
cp ../pmhub-monitor/target/pmhub-monitor.jar  ./pmhub/monitor/jar
```

```
echo ”begin copy pmhub-system ”
cp ../pmhub-modules/pmhub-system/target/pmhub-system.jar
./pmhub/modules/system/jar

echo ”begin copy pmhub-job ”
cp ../pmhub-modules/pmhub-job/target/pmhub-job.jar
./pmhub/modules/job/jar

echo ”begin copy pmhub-gen ”
cp ../pmhub-modules/pmhub-gen/target/pmhub-gen.jar
./pmhub/modules/gen/jar

echo ”begin copy pmhub-project ”
cp ../pmhub-modules/pmhub-project/target/pmhub-project.jar
./pmhub/modules/project/jar

echo ”begin copy pmhub-workflow ”
cp ../pmhub-modules/pmhub-workflow/target/pmhub-workflow.jar
./pmhub/modules/workflow/jar
```

这样就可以将 jar 包合配置文件统一路径，方便管理和一键启动。

开启服务器防火墙

服务器防火墙端口的开启也是基于阿里云安全组开启之后，其实开启命令比较简单，如果是 Debian 系统的话是：

```
ufw allow 9849/tcp
```

如果是 centos 系统：

```
firewall-cmd --add-port=80/tcp --permanent
```

```

但是端口有好多，能不能一次性开启呢？可以的，同样编写 shell 脚本，一键启动即可：

```
sh deploy.sh port
查看是否开启
ufw status
```

启动 Nginx
---

Nginx 我建议不要通过容器来安装，直接安装在宿主机上会比较好。

安装方法如下：

```
更新系统
1、安装
apt install -y nginx
2、看版本号
nginx -v
显示nginx的版本号和编译信息
nginx -V
查看安装的所有 Nginx 包
查看 Nginx 安装相关的文件位置信息
whereis nginx

3、启动Nginx
cd /usr/sbin/
./nginx

4、查询进程：

ps -ef | grep nginx
查看进程id
ps -C nginx -o pid
```
```

5、检查配置是否正确
nginx –t

重启命令：systemctl restart nginx

6、使配置生效

nginx –s reload

```

在 nginx 中配置前段的 dist 文件所在的路径以及域名和 https 即可完成反向代理。意味着你就可以**直接通过域名访问到网站啦**。

一键启动基础环境
---

基础环境包括 MySQL、Redis、Nacos 等，同样构建了 shell 脚本方便一键启动。

```
进去目录
cd /安装路径/pmhub/docker

启动基础环境（必须）
sh deploy.sh base

```

其实内部也就是几个 docker-compose 启动命令：

```
# 启动基础环境（必须）
base(){
docker-compose up -d pmhub-mysql pmhub-redis pmhub-nacos
}

```

一键启动应用服务
---
```

同里，我们将需要一次性启动的几个服务放在一键启动脚本中即可一键启动，但是如果服务器 CPU 核数不足，不建议一次启动过多服务，否则直接就会把服务器搞咋了，推荐还是单个服务启动并观察内存使用情况。

```
docker-compose up -d 你要启动的服务
```

扩展：jar 包启动
==========

如果在本地或者不想通过 docker 的方式启动，也完全可以用 jar 的方式启动，但并不推荐，一来比较慢，二来不方便。

举个例子，需要启动 pmhub-gateway：

```
# 定位到jar目录
cd /你的安装目录/pmhub/docker/pmhub/gateway/jar

# 后台启动
java -jar pmhub-gateway.jar &

# 查看是否启动
netstat -tuln | grep 6880
或者
ps aux | grep pmhub-gateway.jar

# 如何关闭服务
kill 12345
```

至此，在服务器上部署上线服务就已经完成，但还有不少的细节，比如域名如何配置啊，如何进行 ssl 解析啊，如何进行 CI/CD 自动化部署啊，这些其实都需要自己手动去玩了才能搞明白。

如果是在大公司，上线流程其实**还更复杂**，一个优秀的程序员一定是懂的上线的流程的，我想这可能是面试官为什么对未上线项目这么在意的一些原因

吧（个人观点）。

ending
======

历经几个月时间，我们的的首个开源项目就要上线了，这是一套基于 SpringCloud & SpringCloud Alibaba & LLM 的分布式微服务的智能项目管理系统，这个项目旨在让同学们快速掌握微服务/分布式项目的架构设计和开发流程，如果想在校招或者社招中拿到一个**满意的 offer**， 这个项目将是一个非常 nice 的选择。

项目包括认证、流程、项目管理、系统、网关等服务。包含了 Redis 缓存、RocketMQ 消息队列、Docker 容器化、Jenkins 自动化部署、Spring Security 安全框架、Nacos 服务注册和发现、sentinel 分布式事务、Spring Boot Actuator 服务监控、SkyWalking 链路追踪、OAuth2 统一认证、OpenFeign 服务调用，Vue3 前端框架等互联网开发中需要用到的主流技术栈，可以帮助同学们快速掌握微服务/分布式项目的核心知识点。

并且同时也是一套企业工作流的开发框架，您可以根据自身需求，快速定制出适合自己公司的企业工作流系统。

项目目前已经取得** 24 个 star** 了（每一个 star 都格外珍惜）

真希望能尽快和大家见面呀！

原文链接: https://juejin.cn/post/7375928754147754011