

- 1、所需要的组件
- 2、安装Docker
 - 为什么需要安装Docker?
 - Linux安装Docker
- 3、Docker常用指令
- 4、安装MySQL
- 5、Docker安装Redis
- 6、安装Nacos
 - Nacos的作用
 - 单体服务安装
- 7、安装RocketMQ服务
 - MQ的作用
 - RocketMQ的基础服务架构
 - 安装RocketMQ服务
 - 安装dashboard面板服务
 - RocketMQ快速体验
- 8、环境总结

一、基础环境搭建

阶段目标：快速搭建所需的中间件服务。

1、所需要的组件

- 操作系统：Linux
- 数据库：MySQL
- 缓存：Redis
- 配置中心：Nacos
- 消息中间件：RocketMQ

MySQL和Redis对机器资源要求比较少，采用Docker安装。Nacos和RocketMQ对机器资源要求比较高，原生安装。

2、安装Docker

为什么需要安装Docker?

docker官网：<https://docs.docker.com/engine/install/centos/>



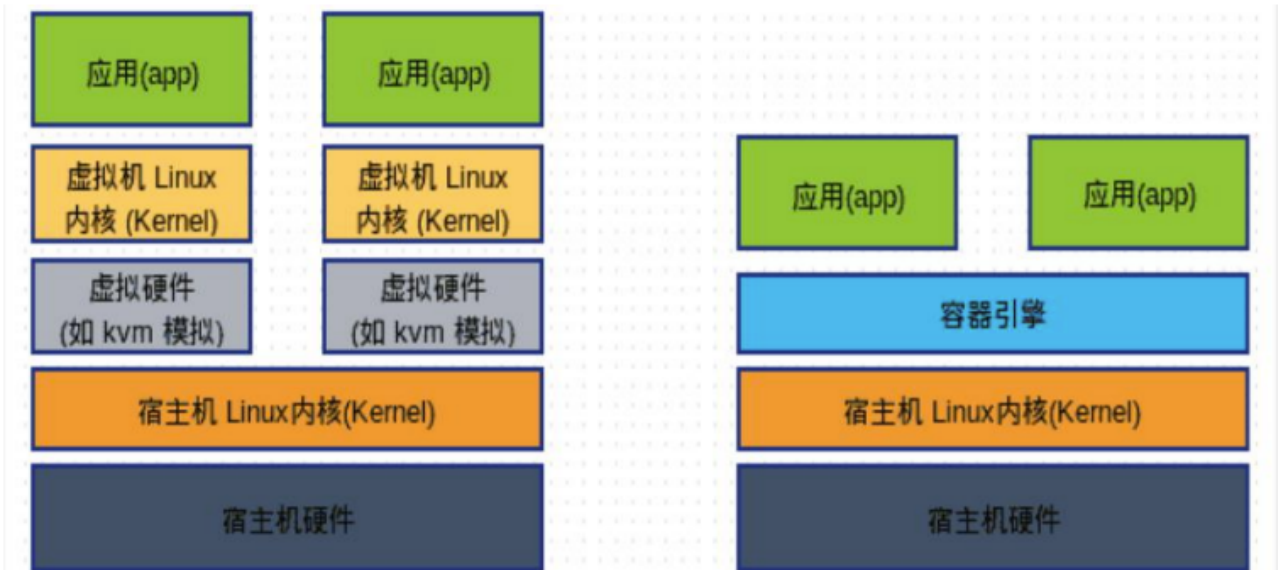
Docker是一个开源的容器化平台，可以帮助开发者将应用程序和其依赖的环境打包成一个可移植、可部署的容器。Docker的主要目标是通过容器化技术实现应用程序的快速部署、可移植性和可扩展性，从而简化应用程序的开发、测试和部署过程。

容器化是一种虚拟化技术，它通过在操作系统层面隔离应用程序和其依赖的运行环境，使得应用程序可以在一个独立的、封闭的环境中运行，而不受底层操作系统和硬件的影响。与传统的虚拟机相比，容器化具有以下优势：

- 轻量级: 容器与宿主机共享操作系统内核，因此容器本身非常轻量级，启动和停止速度快，资源占用少。
- 可移植性: 容器可以在任何支持相应容器运行时的系统上运行，无需关注底层操作系统的差异，提供了高度的可移植性。
- 快速部署: 容器化应用程序可以通过简单的操作进行打包、分发和部署，减少了部署过程的复杂性和时间成本。
- 弹性扩展: 可以根据应用程序的需求快速创建、启动和停止容器实例，实现应用程序的弹性扩展和负载均衡。
- 环境隔离: 每个容器都具有独立的运行环境，容器之间相互隔离，不会相互干扰，提供了更好的安全性和稳定性。

docker和传统虚拟机区别

虚拟机是一个主机模拟出多个主机，需要先拥有独立的系统。传统虚拟机，利用hypervisor，模拟出独立的硬件和系统，在此之上创建应用。docker 是在主机系统中建立多个应用及配套环境，把应用及配套环境独立打包成一个单位，是进程级的隔离。



Linux安装Docker

```
# 1、卸载旧版本
yum remove docker \
docker-client \
docker-client-latest \
docker-common \
docker-latest \
docker-latest-logrotate \
docker-logrotate \
docker-engine
# 2、安装需要的包
yum install -y yum-utils
# 3、设置镜像仓库
yum-config-manager \
--add-repo \
https://download.docker.com/linux/centos/docker-ce.repo
# 默认用的是国外的仓库，推荐使用国内阿里云的仓库
yum-config-manager \
--add-repo \
https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
# 更新yum软件包索引
yum makecache fast
# 4、安装docker.
yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
# 5、检查docker是否安装成功
docker version
# 如果看不到版本，就需要手动启动Docker服务。
# 查看Docker服务状态，如果没有启动，systemctl start docker
systemctl status docker
# 6、跑一个简单镜像
[root@master ~]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:2498fcea14358aa50ead0cc6c19990fc6ff866ce72aeb5546e1d59caac3d0d60f
Status: Downloaded newer image for hello-world:latest
```

Hello from Docker!

This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
(amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

```
$ docker run -it ubuntu bash
```

Share images, automate workflows, and more with a free Docker ID:

<https://hub.docker.com/>

For more examples and ideas, visit:

<https://docs.docker.com/get-started/>

#7、查看容器和镜像

```
docker container ls -a
docker images ls
#8、删除所有容器和镜像
docker container rm $(docker container ls -aq)
docker image rm -f $(docker image ls -q)
#9、卸载docker - 了解
yum remove docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
# 删除docker默认工作目录
rm -rf /var/lib/docker
```

docker安装完成后，建议做一个操作，配置阿里云镜像。这样可以加快镜像下载速度。

登录阿里云平台，进入容器镜像服务，获取镜像加速器地址。



阿里云

🏠 工作台

容器镜像服务

实例列表

制品中心

镜像工具

镜像加速器

容器镜像服务 / 镜像加速器

镜像加速器

⚠️ 由于当前运营商网络问题，可能会导致您拉取 Docker Hub 镜像变慢。建议您手动拉取镜像到本地。

加速器

加速器地址

https://qquu7r94.mirror.aliyuncs.com 复制

操作文档

Ubuntu CentOS Mac Windows

1. 安装 / 升级Docker客户端

推荐安装 1.10.0 以上版本的Docker客户端，参考文档[docker-ce](#)

2. 配置镜像加速器

针对Docker客户端版本大于 1.10.0 的用户

您可以通过修改daemon配置文件 `/etc/docker/daemon.json` 来使用加速器

```
sudo mkdir -p /etc/docker
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors": ["https://qquu7r94.mirror.aliyuncs.com"]
}
EOF
sudo systemctl daemon-reload
sudo systemctl restart docker
```

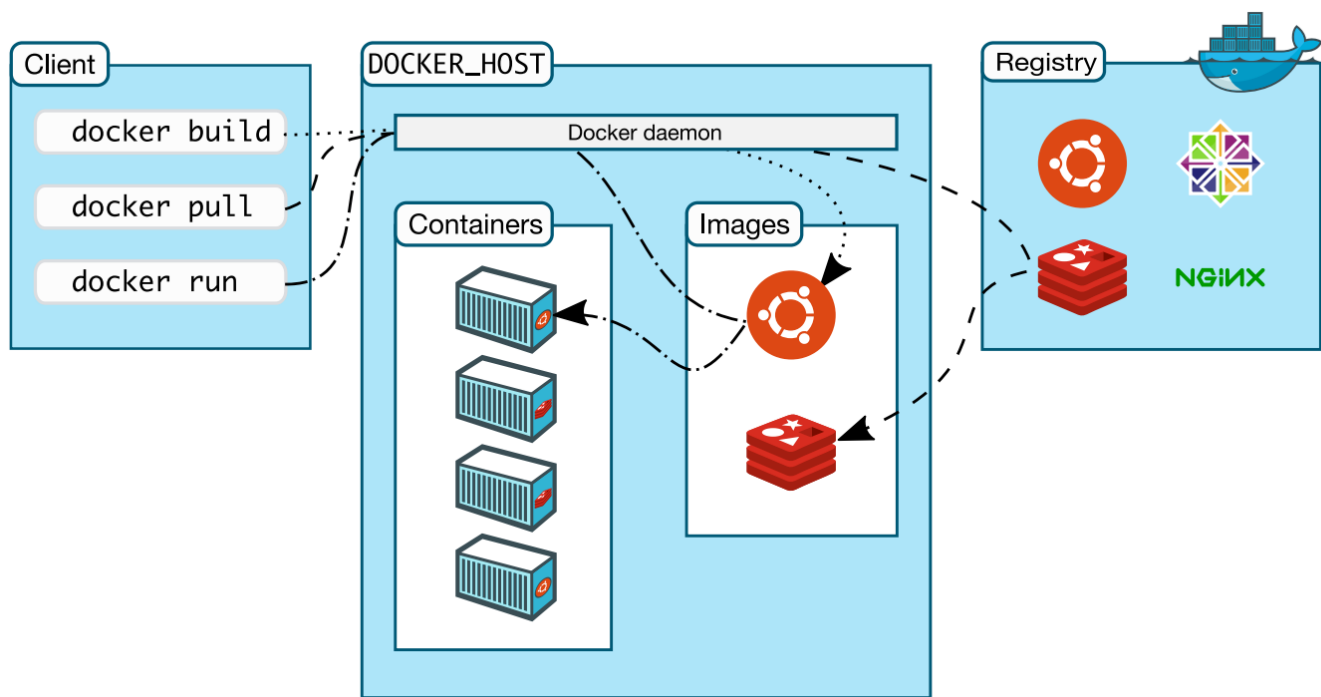
Docker配置加速镜像：

```
sudo mkdir -p /etc/docker
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors": ["https://qquu7r94.mirror.aliyuncs.com"]
}
EOF
sudo systemctl daemon-reload
sudo systemctl restart docker
```

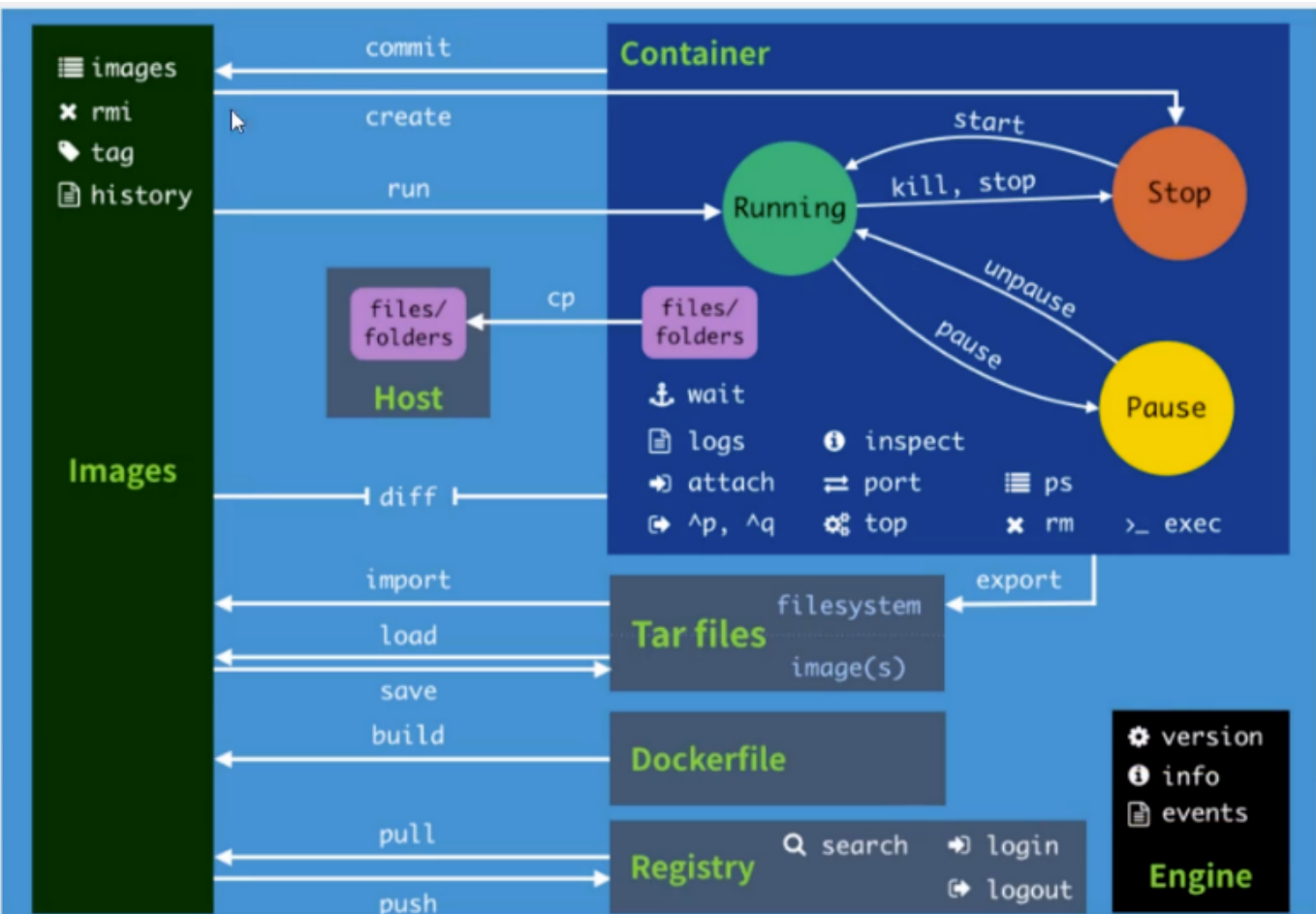
地址替换成你自己的加速器地址

3、Docker常用指令

Docker的基础架构：



Docker常用指令汇总：



Docker实操：

```
# 帮助命令
docker version      # docker版本
docker info         # docker系统信息
docker xxx --help   #查看帮助

# 镜像操作
docker images # docker image ls
docker search nginx # 搜索镜像
docker pull nginx # 下载镜像 docker image pull
docker inspect nginx # 查看镜像信息

docker run -d --name mynginx -p 80:80 nginx # 启动镜像
docker ps # 查看镜像执行情况
docker exec -it {container_id} /bin/bash

docker rmi 镜像ID # 删除镜像
docker image rm -f $(docker image ls -q) #删除所有镜像

# 容器操作
docker ps
docker container ls -a #查询所有容器
docker rm $(docker ps -aq) #删除所有容器
```

4、安装MySQL

dockerhub参考地址：https://hub.docker.com/_/mysql

需要特别注意的是，对于镜像中需要持久化保存的文件，需要通过-v挂载到宿主机上，这样这些文件才不会随着容器关闭而消失。

对于MySQL，需要将他的日志文件、数据文件和配置文件挂载到宿主机上。

```
# 拉取镜像
docker pull mysql:8
# 启动mysql并配置工作目录。将容器中的工作目录挂载到本机，这样这些数据文件才能持久化保存。
docker run -p 3306:3306 --name mysql8 -v /app/docker/mysql/log:/var/log/mysql -v
/app/docker/mysql/data:/var/lib/mysql -v /app/docker/mysql/conf:/etc/mysql -v
/app/docker/mysql/mysql-files:/var/lib/mysql-files -e MYSQL_ROOT_PASSWORD=root -d mysql:8
# MySQL默认只能在本机登录，也就是只能从容器内登录。需要调整权限，允许远程访问。
docker container ls    #获取容器ID
# 进入容器
docker exec -it 29387949bc43 /bin/bash
# 在容器内登录MySQL 。 不要输密码
mysql -u root
# 调整MySQL，允许远程连接
mysql> use mysql;
mysql> grant all PRIVILEGES on *.* to root@ '%' WITH GRANT OPTION;
mysql> update user set host= '%' where user='root';
mysql> ALTER user 'root'@ '%' IDENTIFIED BY 'root' PASSWORD EXPIRE NEVER;
mysql> ALTER user 'root'@ '%' IDENTIFIED WITH mysql_native_password BY 'root';
mysql> flush privileges;
# 如果都正常执行，那么MySQL服务就可以用客户端工具远程登录了。
```

开发阶段，只搭建单机服务即可。如果需要高可用，可以搭建MySQL的集群。

5、Docker安装Redis

同样，需要注意将Redis的配置文件和日志文件挂载到宿主机上。

```
docker pull redis:latest
# 将Redis的配置文件和数据文件挂载到宿主机上
docker run -p 6379:6379 --name redis -v
/Users/roykingw/docker/redis/config:/etc/redis.conf -v
/Users/roykingw/docker/redis/data:/app/redis -d redis redis-server /etc/redis.conf
# 如果正常执行，那么Redis的服务就可以用客户端工具远程登录了。
```

6、安装Nacos

Nacos的作用

- 服务注册中心
- 配置中心

单体服务安装

前置安装Java，过程略。

下载页面：<https://nacos.io/download/nacos-server/>。这次选择下载2.2.0版本

安装Nacos之前，需要注意一下。Nacos默认将元数据记录到derby内存数据库中，容易丢失。所以通常需要将元数据改为存储到MySQL中。

使用MySQL保存元数据，需要在MySQL中创建独立的库和表。相关表的初始化脚本记录在nacos的conf/mysql-schema.sql文件中。

接下来需要修改nacos的配置文件，主要调整以下部分：

```
### If use MySQL as datasource:
### Deprecated configuration property, it is recommended to use `spring.sql.init.platform`
replaced.
# spring.datasource.platform=mysql
spring.sql.init.platform=mysql

### Count of DB:
db.num=1

### Connect URL of DB:
db.url.0=jdbc:mysql://127.0.0.1:3306/nacos?
characterEncoding=utf8&connectTimeout=1000&socketTimeout=3000&autoReconnect=true&useUnicod
e=true&useSSL=false&serverTimezone=UTC
db.user.0=root
db.password.0=root
```

接下来就可以启动了。单机模式启动Nacos需要加上参数 `-m standalone`

```
bin/startup.sh -m standalone
```

如果启动没有问题，就可以访问nacos页面了。访问地址：<http://192.168.65.210:8848/nacos>。默认用户名和密码都是nacos

在nacos中，有个命名空间的概念。通过命名空间可以在不同应用之间形成数据隔离。

接下来，我们可以在nacos中创建一个命名空间，名为tl-live，作为后续项目的数据空间。

新建命名空间

命名空间ID(不填
则自动生成)

tl-live

* 命名空间名

tl-live

* 描述

仿抖音直播平台

确定

取消

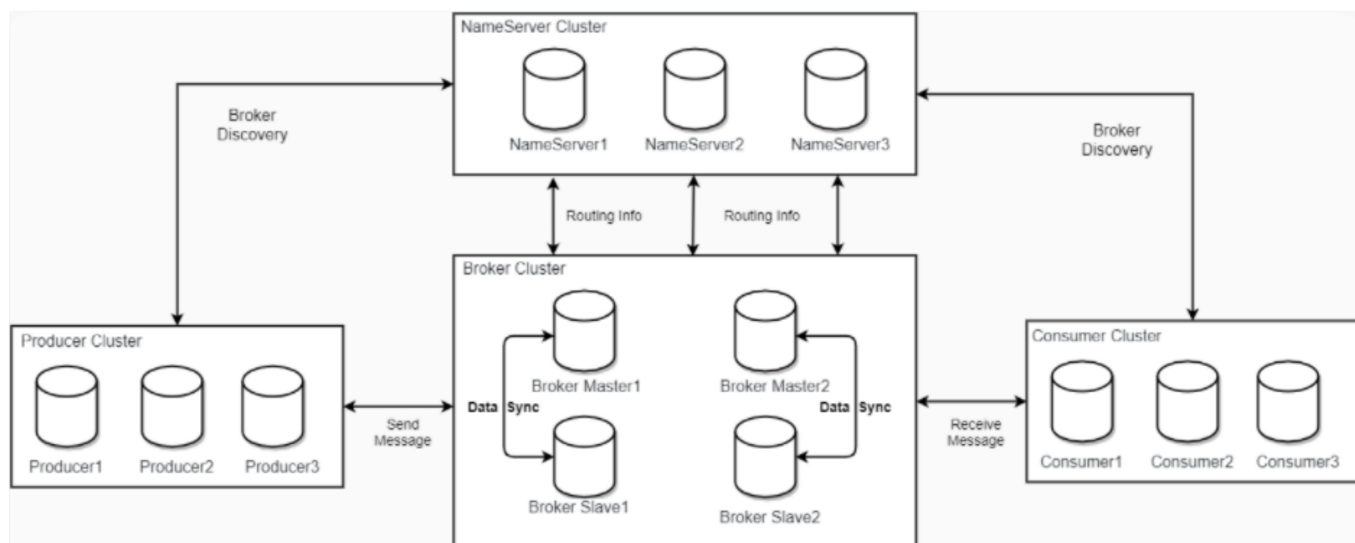
7、安装RocketMQ服务

MQ的作用

- 异步 送快递
- 解耦 翻译
- 削峰 三峡大坝

任何关于MQ应用场景的面试题，都往这三个方面靠。

RocketMQ的基础服务架构



安装RocketMQ服务

下载地址：<https://rocketmq.apache.org/download/> 选择5.2.0版本。下载 rocketmq-all-5.2.0-bin-release.zip 和 rocketmq-dashboard-master.zip 两个组件。

rocketmq-all-5.2.0-bin-release.zip解压后，可以直接启动nameserver服务

```
cd /app/rocketmq_5.2.0
nohup bin/mqnamesrv &
```

启动后，会在当前目录生成一个nohup.out日志文件。观察日志文件，有下一行关键日志，表示服务启动成功。

```
The Name Server boot success. serializeType=JSON, address 0.0.0.0:9876
```

接下来启动broker服务。启动之前建议对配置文件做一些调整。vi conf/broker.conf，增加下面内容。主要是对存储文件进行规划，便于后续掌握服务运行情况。

```
#允许自动创建topic 用于测试
autoCreateTopicEnable=true
#存储路径
storePathRootDir=/app/rocketmq/store
storePathCommitLog=/app/rocketmq/store/commitlog
storePathConsumeQueue=/app/rocketmq/store/consumequeue
storePathIndex=/app/rocketmq/store/index
storeCheckpoint=/app/rocketmq/store/checkpoint
abortFile=/app/rocketmq/store/abort
```

然后，需要配置一个环境变量，指向name server服务地址

```
vi ~/.bash_profile
# 增加下面配置项
# export NAMESRV_ADDR=192.168.65.210:9876
# 增加完成后, 让配置文件生效
source ~/.bash_profile
```

启动broker服务

```
cd /app/rocketmq_5.2.0
nohup bin/mqbroker -c conf/broker.conf &
# 启动成功的关键日志
# The broker[broker-a, 192.168.65.210:10911] boot success. serializeType=JSON and name
server is 192.168.65.210:9876
```

安装dashboard面板服务

RocketMQ提供了一个基于Web的管理服务Dashboard，可以基于浏览器快速监控并管理RocketMQ服务。

rocketmq-dashboard-master.zip包中，只包含了Dashboard服务的源码，并没有直接提供编译后的jar包。需要在本地开发环境中安装Maven，自行进行编译。

Maven安装过程略

编译指令：

```
mvn clean package -Dmaven.test.skip=true
```

编译完成后，在源码的target目录下可以获得可运行的jar包rocketmq-dashboard-1.0.1-SNAPSHOT.jar。

接下来把jar包上传到服务器上，并在jar包同目录下创建一个文件application.yml，配置dashboard服务指向的nameserver服务地址

```
rocketmq:
  config:
    namesrvAddrs:
      - 192.168.65.210:9876
```

到此就可以启动dashboard服务了

```
nohup java -jar rocketmq-dashboard-1.0.1-SNAPSHOT.jar &
```

如果nohup.out中没有报错信息，那么就可以访问dashboard的管理页面。

在集群中可以看到当前服务状况。

← → ↺ ⚠ 不安全 192.168.65.210:8080/#/cluster

🔍 ☆ 🟢 🔴 ⚡ 🗑 👤 ⋮

RocketMQ仪表板

运维

驾驶舱

集群

主题

消费者

生产者

消息

死信消息

消息轨迹

更换语言

集群:

DefaultCluster

分片	编号	地址	版本	生产消息TPS	消费消息TPS	昨日生产总数	昨日消费总数	今天生产总数	今天消费总数	操作
broker-a	0(master)	192.168.65.210:10911	V5_2_0	0.00		0	0	0	0	<div>状态</div> <div>配置</div>

RocketMQ快速体验

RocketMQ内置了测试案例，可以快速体验RocketMQ的收发消息功能。

```
cd /app/rocketmq_5.2.0
# 发送消息
bin/tools.sh org.apache.rocketmq.example.quickstart.Producer
# 接受消息
bin/tools.sh org.apache.rocketmq.example.quickstart.Consumer
```

很显然，tools.sh实际上是提供了一个RocketMQ客户端的运行环境，然后去执行对应的java类。如果想要了解在Java应用中如何往RocketMQ收发消息，可以去查看下对应的测试类。

8、环境总结

开发阶段，我们的目的是为了尽快启动开发工作，所以所有服务都采用单体方式快速安装。在实际项目中，这些中间件服务，都会采用集群化的方式进行部署。如果你想要深入了解这些中间件，可以来学习我们的VIP课程。