# Docker 安装mysql5.7

## 镜像拉取

docker pull mysql:5.7

## 启动容器 设置卷，编码chartset

sudo docker run -p 3306:3306 --name fap-mysql -v /root/mysql/conf:/etc/mysql -v /root/mysql/logs:/var/log/mysql -v /root/mysql/data:/var/lib/mysql -e MYSQL\_ROOT\_PASSWORD=123456 -d mysql:5.7 --character-set-server=utf8mb4 --collation-server=utf8mb4\_unicode\_ci

## 创建数据库，默认便阿门utf8mb4

CREATE DATABASE `FapCore`

DEFAULT CHARACTER SET utf8mb4

DEFAULT COLLATE utf8mb4\_general\_ci

## 导入脚本

修改参数：max\_allowed\_packet

1.可以编辑my.cnf,在[mysqld]段或者mysql的server配置段进行修改。

max\_allowed\_packet = 100M

修改my.cnf,配置要重载才能生效

2.

设置 show VARIABLES like '%max\_allowed\_packet%';

增加最大允许包的大小，否则会insert失败

set global max\_allowed\_packet = 2\*1024\*1024\*10

exit

重新进入 mysql -uroot -proot

设置编码

show variables like 'character%';

set names utf8mb4;

Rz上传脚本

进入容器

docker exec -it mysql bash

进入mysql -u用户名 -p密码

mysql -uroot -proot

执行脚本文件source命令

source /var/lib/mysql/MYSQL.sql

mapping : root/mysql/data

## 设置编码

set names utf8mb4;

# docker 安装 mssql2017

## 拉取镜像

sudo docker pull mcr.microsoft.com/mssql/server:2019-CTP3.2-ubuntu

## 运行容器

sudo docker run -e "ACCEPT\_EULA=Y" -e "MSSQL\_SA\_PASSWORD=m,./1234" --name "mssql" -p 1433:1433 -v /var/opt/mssql:/var/opt/mssql -d mcr.microsoft.com/mssql/server:2017-latest

#create directory for backup in container[fapcore]

sudo docker exec -it fapcore mkdir /var/opt/mssql/backup

#from host copy .bak file to contianer

sudo docker cp database/fapcoredb.bak fapcore:/var/opt/mssql/backup

#还原数据库

sudo docker exec -it fapcore /opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P "m,./1234" -Q "RESTORE DATABASE FapCore FROM DISK = '/var/opt/mssql/backup/fapcoredb.bak' WITH MOVE 'Fap' TO '/var/opt/mssql/data/FapCore.mdf',MOVE 'Fap\_log' TO '/var/opt/mssql/data/FapCore\_log.ldf'"

#执行脚本

#更新附件存储路径

sudo docker exec -it fapcore /opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P 'm,./1234' -Q "update fapcore.dbo.fapconfig set paramvalue='/var/fapcore/annex' where paramkey='file.directory.path'"

#设置Clr

sudo docker exec -it fapcore /opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P 'm,./1234' -Q "exec sp\_configure 'show advanced options', '1';reconfigure;exec sp\_configure 'clr enabled', '1'; reconfigure;exec sp\_configure 'show advanced options', '1';ALTER DATABASE FapCore SET TRUSTWORTHY on;exec sp\_changedbowner 'sa'"

# 安装Jenkins

下载链接<http://mirrors.jenkins.io/war-stable/>，下载最新版

cd /var/lib/jenkins-dotnet-core

wget [http:*//mirrors.jenkins.io/war-stable/2.107.2/jenkins.war*](http://mirrors.jenkins.io/war-stable/2.107.2/jenkins.war)*;下载，上面最新url替换这个地址*

#切换文件夹

cd /var/lib

#新建一个jenkin-home文件夹

mkdir Jenkins-home

#设置环境变量

export JENKINS\_HOME=/var/lib/Jenkins-home

#启动一个新会话

tmux;

#切换到war文件夹

cd Jenkins-dotnet-core

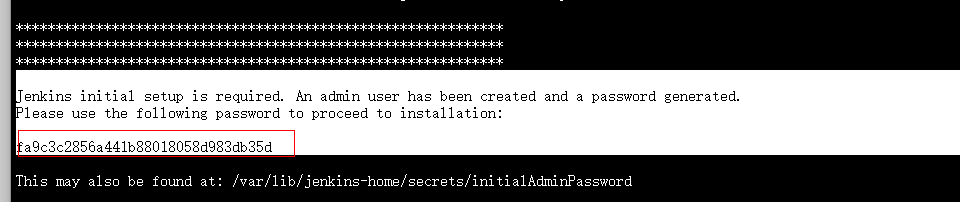
#执行jenkins

java -jar Jenkins.war

启动jenkins

首页输入密码 如下：

fa9c3c2856a441b88018058d983db35d



安装推荐插件

# Docker部署Nginx

## 拉取镜像

docker pull nginx

## 启动容器

建立文件夹

mkdir -p /data/nginx/conf

mkdir -p /data/nginx/conf.d

mkdir -p /data/nginx/html

mkdir -p /data/nginx/logs

运行 docker run -it nginx bash

复制容器中的 nginx.conf

docker cp 05416d52be79:/etc/nginx/nginx.conf /data/nginx/conf

复制容器中的default.conf

docker cp 05416d52be79:/etc/nginx/conf.d/default.conf /data/nginx/conf.d

运行容器

docker run --name mynginx -d -p 80:80 -v /data/nginx/html:/usr/share/nginx/html -v /data/nginx/conf/nginx.conf:/etc/nginx/nginx.conf -v /data/nginx/conf.d:/etc/nginx/conf.d -v /data/nginx/logs:/var/log/nginx nginx

## 配置反向代理

在conf.d里面用 vi hrsoft.club.conf 命令创建域名配置文件并输入以下内容：

server {

listen 80;

server\_name hrsoft.club \*. hrsoft.club;

location / {

proxy\_pass http://localhost:5000;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection keep-alive;

proxy\_set\_header Host $host;

proxy\_cache\_bypass $http\_upgrade;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto $scheme;

}

}

注意：如果nginx是容器部署的话，proxy\_pass地址必须是另一个容器中的部署ip，不能是localhost，因为localhost是nginx容器自身ip

# Xshell 上传下载文件

## 安装

apt install lrzsz

## 命令

上传（rz）,下载(sz)

# ASP.NET Core Kestrel免费实现https

只要给我们的web服务器配置一个证书就行了，证书可以买，也可以用免费的Let's Encrypt，此证书提供商是多个牛X大公司为了推进全球https化搞出来的，所以不用担心免费的会有啥问题。唯一的问题是90天有效期，所以你得提前免费续签，当然有办法容易的实现自动续签。

为了实现自动化的证书申请、发放、续签、删除.....各种证书相关的功能，所以Let's Encrypt提出了个ACME协议，不同编程语言都可以实现这个协议来完成这些工作。

## 添加nuget包

paket add **McMaster.AspNetCore.LetsEncrypt** --version 0.3.0

<https://github.com/natemcmaster/LetsEncrypt>

## 启动配置

using Microsoft.Extensions.DependencyInjection;

public class Startup

{

public void ConfigureServices(IServiceCollection services)

{

services.AddLetsEncrypt();

}

}

## 配置文件

// appsettings.json

{

"LetsEncrypt": {

// Set this to automatically accept Let's Encrypt's terms of service.

// If you don't set this in config, you will need to press "y" whenever the application starts

"AcceptTermsOfService": true,

// You must at least one domain name

"DomainNames": [ "example.com", "www.example.com" ],

// You must specify an email address to register with letsencrypt.org

"EmailAddress": "it-admin@example.com"

}

}