编程知识

Using docker to build nestjs and Mongo development environment

2021-08-04 17:50:50 / BY LEOOOY

brief introduction

This article describes how to use `Docker` build `NestJS` and `MongoDB` Development environment of .

Full project address : [github.com/LeoooY/nest...]

To successfully run the code , And understand how each part works , You need at least a basic understanding of the following knowledge points :

- `JavaScript` & `TypeScript`
- `Docker` Basic use of
- `MongoDB` Basic use of

precondition

- install `Docker` and `Docker-Compose`
- install `NodeJS`

Begin to build

Main steps :

1. establish `NestJS` project
2. Containerization `NestJS` project
3. Use `Docker-Compose` Choreography container
4. stay `NestJS` Connection in project `MongoDB` service

establish `NestJS` project

Introduction | NestJS

```
$ npm i -g @nestjs/cli
$ nest new project-name
 Copy code
```

Use `nest new` After the command is created , `cd` To `project-name` Project root execution `start:dev` .

Open browser access `http://localhost:3000` , Normal access means `NestJS` The project has been created .

Containerization `NestJS` project

In this step , We need the just created `NestJS` The project is packaged into a `Docker` Containers .

1. First, in the `NestJS` Create a new project root directory `Dockerfile` file

```
$ touch Dockerfile
 Copy code
```

2. modify `Dockerfile` The contents of the document

```

# Docker Multistage construction

### DEV Environmental Science  ###
FROM node:14.17.3 AS development

#  Navigate to the container working directory
WORKDIR /usr/src/app
#  Copy package.json
COPY package*.json ./

RUN npm install glob rimraf
RUN npm install --only=development
COPY . .
RUN npm run build


### PROD Environmental Science  ###
FROM node:14.17.3 as production

ARG NODE_ENV=production
ENV NODE_ENV=${NODE_ENV}

WORKDIR /usr/src/app

COPY package*.json ./

RUN \
  npm config set registry https://registry.npm.taobao.org \
  && npm install --only=production

COPY . .

COPY --from=development /usr/src/app/dist ./dist

CMD ["node", "dist/main"]
 Copy code
```

Such a `NestJS` Project `Docker` The image is customized .

> Because there is still `MongoDB` Related container services , We don't use... Directly `Docker` Command to build 、 function , But use `Docker-Compose` Choreography container

Use `Docker-Compose` Choreography container

`Docker-Compose` Of `docker-compose.yml` A configuration file can define a set of associated application containers as a project , In this way, we can easily manage `NestJS` and `MongoDB` Service for .

1. stay `NestJS` Create a new project root directory `docker-compose.yml` file

```
$ touch docker-compose.yml
 Copy code
```

2. modify `docker-compose.yml` The contents of the document

```
version: '3.9'
services:
  dev:
    container_name: server-dev
    image: server-dev:1.0.0
    build:
      context: .
      target: development
      dockerfile: ./Dockerfile
    command: npm run start:debug
    ports:
      - 3000:3000
      - 9229:9229
    networks:
      - server-network
    volumes:
      - .:/usr/src/app
      - /usr/src/app/node_modules
    restart: unless-stopped
    environment:
      MONGO_URL: mongodb
  prod:
    container_name: server-prod
    image: server-prod:1.0.0
    build:
      context: .
      target: production
      dockerfile: ./Dockerfile
    command: npm run start:prod
    ports:
      - 3000:3000
      - 9229:9229
    networks:
      - server-network
    volumes:
      - .:/usr/src/app
      - /usr/src/app/node_modules
    restart: unless-stopped
  mongodb:
    image: mongo:5.0.0
    container_name: server-mongodb
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=pass12345
    volumes:
      - mongodb-data:/data/db
    networks:
      - server-network
    ports:
      - 27017:27017
    healthcheck:
      test: echo 'db.runCommand("ping").ok' | mongo localhost:27017/test --quiet
      interval: 30s
      timeout: 10s
      retries: 3
    restart: unless-stopped
  mongo-express:
    image: mongo-express
    container_name: server-mongo-express
    environment:
      - ME_CONFIG_MONGODB_SERVER=mongodb
      - ME_CONFIG_MONGODB_ENABLE_ADMIN=true
      - ME_CONFIG_MONGODB_ADMINUSERNAME=root
      - ME_CONFIG_MONGODB_ADMINPASSWORD=pass12345
      - ME_CONFIG_BASICAUTH_USERNAME=admin
      - ME_CONFIG_BASICAUTH_PASSWORD=admin123
    volumes:
      - mongodb-data
    depends_on:
      - mongodb
    networks:
      - server-network
    ports:
      - 8081:8081
    healthcheck:
      test: wget --quiet --tries=3 --spider http://admin:admin123@localhost:8081 || exit 1
      interval: 30s
      timeout: 10s
      retries: 3
    restart: unless-stopped
volumes:
  mongodb-data:
    name: mongodb-data
networks:
  server-network:
 Copy code
```

To this step , We have completed all the steps of containerization , The rest is in `NestJS` De link `MongoDB` service .

stay `NestJS` Connection in project `MongoDB` service

We use `NestJS` The recommended `@nestjs/mongoose` Tools to connect `MongoDB` service .

1. install `@nestjs/mongoose`

```
$ npm install --save @nestjs/mongoose mongoose
# or yarn
$ yarn add -D @nestjs/mongoose mongoose
 Copy code
```

2. Connect `MongoDB` service

`app.module.ts`

```
import { Module } from '@nestjs/common';
import { AppController } from './app.controller';
import { AppService } from './app.service';
import { MongooseModule } from '@nestjs/mongoose';

const url = process.env.MONGO_URL || 'localhost';

@Module({
  controllers: [AppController],
  providers: [AppService],
  imports: [
    MongooseModule.forRoot(
      `mongodb://${url}:27017?serverSelectionTimeoutMS=2000&authSource=admin`,
    ),
  ],
})
export class AppModule {}
 Copy code
```

> It should be noted that we use `mongodb` Connection address `process.env.MONGO_URL` yes `docker-compose.yml` As defined in `mongodb` Address of service , Reference resources Accessing a docker container from another container.

Start project

Now? , We have finished all the configuration work , You can run the project .

start-up `NestJS` service 、 `Mongo` Service and `Mongo-Express` service .

```
$ docker-compose up -d dev mongodb mongo-express
 Copy code
```

> Be careful ： When you use `npm` Installed a new `package` when , Need to use `-V` Parameter to recreate the container `node_modules` Anonymous data volume .

```
$ docker-compose up -d -V dev
 Copy code
```

Check the status of each container

```
$ docker ps
 Copy code
```

View the container's log

```
$ docker logs server-dev
$ docker logs server-dev -f # -f  Used for parameter continuous output logs
 Copy code
```

Into the container `shell`

```
$ docker exec -it server-mongodb bash #  Get into mongo Containers
 Copy code
```

Reference material

- Setting up a NestJS project with Docker for Back-End development
- Containerized development with NestJS and Docker
- How To Run MongoDB as a Docker Container
- Containerize Nest.js+MongoDB application in 5 minutes
- Accessing a docker container from another container
- From inside of a Docker container, how do I connect to the localhost of the machine?
- Docker From introduction to practice
- Eggjs Dockerfile
- `@nestjs/mongoose`
- Docker-compose: node_modules not present in a volume after npm install succeeds
- Top 4 Tactics To Keep Node.js Rockin' in Docker

Tags USING DOCKER BUILD NESTJS MONGO

推荐

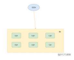
Oracle 12C RAC modifying scan configuration

Oracle modify time zone

In depth study on the relationship between Oracle DB server system time modification and SCN

Oracle block SCN / commit SCN / cleanup SCN description

The first bullet in Oracle ASM Translation Series: Basics ASM Au, extensions, mirroring and failgroups



Every programmer should learn Maven knowledge

sleepless

Jia Ling became the highest box office female director in the world, and the film moved countless audiences!

Women over the age of 50 wear less long pants in summer and learn from mother Miki. It's elegant and fashionable nginx学习

SpringBoot-Elasticsearch

SpringBoot-Kotlin

docker-compose入坑

docker 入坑

docker部署Springboot项目

Linux误操作777之后的恢复

SpringSecurity 入门（一）

SpringSecurity 入门 （三）

SpringSecurity 入门（二）

SpringSecurity 入门 （四）

部署maven私服

Git入坑

SpringSecurity入坑（三）

SpringSecurity入坑（二）

SpringSecurity入坑（五）

Nginx learning

SpringBoot-Elasticsearch

SpringBoot-Kotlin

Docker compose pit entry

Docker pit entry

Docker deploying springboot project

5 个可以加速开发的 VueUse 库函数

Recovery after Linux misoperation 777

Getting started with spring security (1)

Introduction to spring security (3)

Introduction to spring security (2)

Getting started with spring security (4)

Deploy Maven private server

Git pit entry

Spring security pit entry (III)

Spring security pit entry (II)

Spring security pit entry (V)

5 vueuse library functions that can accelerate development

Linux 汇总

Linux summary

网站前端性能优化终极指南

Ultimate Guide to website front end performance optimization

Android development studio beginner tutorial

Compose | to understand the magical modifier

Binder principle analysis for Android Application Engineers

Gradle plug-in series (I) - configure your own gradle plug-in

如何决定响应式网站的 CSS 单位?

Github1.3万星，迅猛发展的JAX对比TensorFlow、PyTorch

How to determine the CSS unit of a responsive website?

GitHub 13000 stars, rapidly developing Jax, compared with tensorflow and pytorch

【每日算法】算法复习一

[daily algorithm] algorithm review I

数据结构算法学习之队列(数组模拟java实现)

Yu Zheng: there must be some people who don't like the spring of jade mansions, such as those with low education and those who like to be picky

Queue for data structure algorithm learning (array simulation Java implementation)

免责声明

本站以网络数据为基准，引入优质的垂直领域内容。本站内容仅代表作者观点，与本站立场无关，本站不对其真实合法性负责。 如有内容侵犯了您的权益，请告知，本站将及时删除。联系邮箱：chxpostbox@gmail.com

友情链接