The Way of Dockerize a Spring Boot and MySQL Application With Docker Compose.

Amila Iroshan · Follow
Published in The Fresh Writes
6 min read · Jan 27, 2023

Pre-requisite,

— Basic knowledge of docker and java with spring boot

— Setup Docker and Docker compose to local machine

Technology Stack

+ Spring Boot 3.0.0-RELEASE

+ Spring Data JPA

+ MySQL — 8.0

+ Docker — version 20.10.21

+ Docker-Compose — version v2.13.0

You can get instruction about docker installation from https://docs.docker.com/desktop/install/windows-install/.I have downloaded docker desktop to my local pc and It is wrapped up docker with docker compose. So no need to install docker compose separately.

Overview

My sample application provide GET api for display list of person's names. The sample data is fetching from MySql DB.

What is :

Docker : Docker is open source containerization platform used for building, packaging, and managing applications in an isolated environment.
Dockerfile : It is the place where we config the model of our docker container. By using dockerfile we can create docker image.
Docker Image: The blueprint for create docker contaiers.(According to oop concepts it is like a class)
Docker Container : It is runnable instance of image.(According to oop concepts it is like a object which is derived from class)
Docker Compose : Docker compose is a tool which helps us to easily handle multiple containers at once.

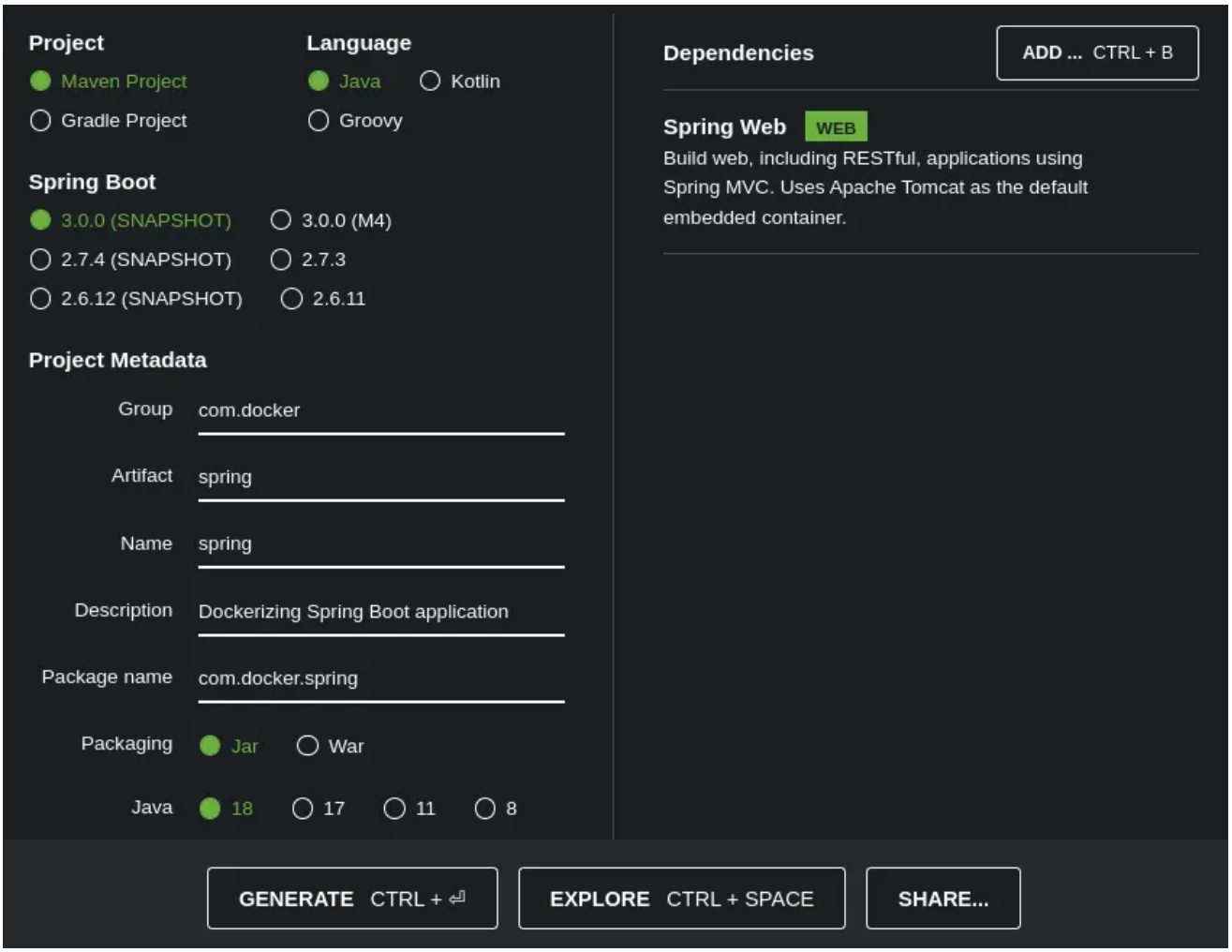
I'm going to following below steps:
1).Create spring boot application and connect it with MySql DB.
2).Create Dockerfile to Spring boot application.
3).Create docker compose configuration file
4).Run the system and inspect running containers

Step 1:

Create spring boot application and connect it with MySql DB.
1).Navigate to https://start.spring.io.

2).Choose

either Gradle or Maven as build tool. In here I'm using maven, Java 18 and .jar as packaging.



Create Spring boot application

3).Click Dependencies and select spring starter web, spring data jpa and mysql connector. This is my pom.xml file.

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2(
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/x
 <modelVersion>4.0.0</modelVersion>
 <parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>3.0.0</version>
  <relativePath/> <!-- lookup parent from repository -->
 </parent>
 <groupId>com.example</groupId>
 <artifactId>basic</artifactId>
 <version>0.0.1-SNAPSHOT</version>
 <name>basic</name>
 <description>Demo project for Spring Boot</description>
 <properties>
  <maven.compiler.source>17</maven.compiler.source>
```

```
        <maven.compiler.target>17</maven.compiler.target>
    </properties>
    <dependencies>
     <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
     </dependency>

      <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter-data-jpa</artifactId>
           </dependency>

           <!-- MySQL -->
           <dependency>
               <groupId>mysql</groupId>
               <artifactId>mysql-connector-java</artifactId>
           </dependency>



    </dependencies>

    <build>
     <plugins>
      <plugin>
       <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
     </plugins>
     <finalName>spring_rest_docker</finalName>
    </build>

   </project>
```

4).Download the resulting ZIP file, which is an archive of a web application that is configured with your choices.
5).Create a GET endpoint to fetch data from db.

```
@RestController
public class BasicController {

 @Autowired
 private PersonService personService;
 @GetMapping("/all")
 public List<Persons> getAll() {
  return personService.findAll();
 }
}
```

6).Connect application with MySql db. Here is my application.properties file.

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/basics?allowPublicKeyRetrieval=
spring.datasource.username=amila_one
spring.datasource.password=Amila_pw

spring.sql.init.mode=always
spring.datasource.initialization-mode=always
```

Step 2:

Create a Dockerfile to Spring boot application. The dockerfile should be in the class path.

```
#
# Build stage
#
FROM maven:3.8.3-openjdk-17 AS build
COPY src /home/app/src
COPY pom.xml /home/app
RUN mvn -f /home/app/pom.xml clean package
EXPOSE 8080
ENTRYPOINT ["java","-jar","/home/app/target/spring_rest_docker.jar"]
```

FROM : Fetching latest version of Java image with maven. This pre define docker image exists on docker hub.
COPY : Copying Project src folder to openjdk-17 container's root directory /home/app/

src.Copy again pom.xml file to /home/app/.
RUN : Execute the mavean command to build the .jar file accoring to given pom.xml file.
EXPOSE : Specify that expose server port
ENTRYPOINT : Execute command for run the .jar file.We can use **CMD** instead of **ENTRYPOINT**.If we use **CMD** we can provide arguments to image when build it.

Step 3:

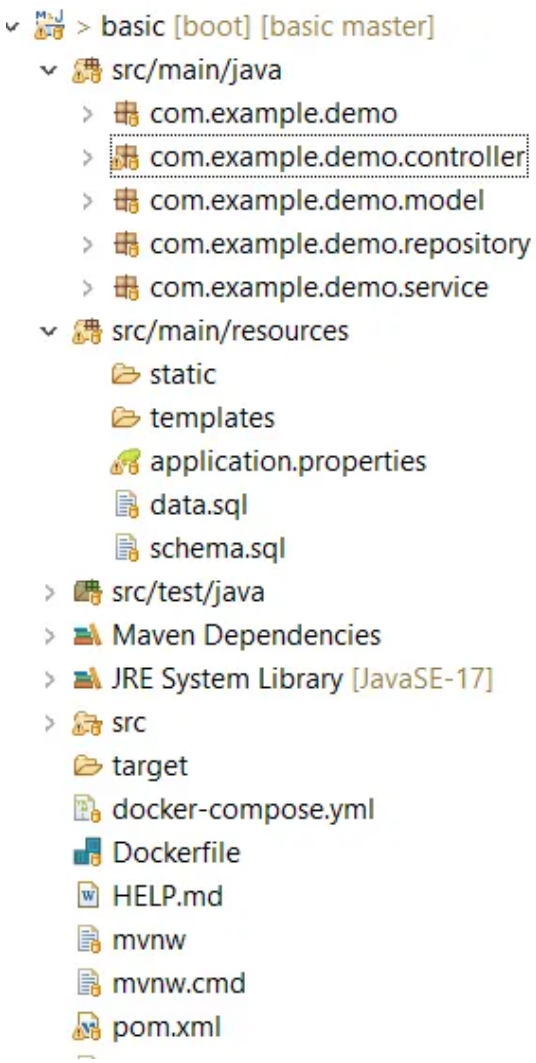
Create docker compose configuration file.The naming convention of this file should be docker-compose.yaml or .yml. This file should be on the class path. This docker compose file helps us to combine the spring boot app and MySql db setup.

```
version: "3.7"
services:
  api_service:
    build: .
    restart: always
    ports:
      - 8080:8080
    networks:
      - springapimysql-net
    environment:
      - spring.datasource.url=jdbc:mysql://mysqldb:3306/basics?allowPublicKeyRetr
    depends_on:
      - mysqldb

    volumes:
      - .m2:/root/.m2

  mysqldb:
    image: "mysql:8.0"
    restart: always
    ports:
      - 3306:3306
    networks:
      - springapimysql-net
    environment:
      MYSQL_DATABASE: basics
      MYSQL_USER: amila_one
      MYSQL_PASSWORD: Amila_pw
      MYSQL_ROOT_PASSWORD: Amila_Rpw
networks:
  springapimysql-net:
```

version: Version of Docker Compose file format.
services: My application has two services: app (Spring Boot) and mysqldb (MySQL

database image).
build: Configuration options that are applied at build time that we defined in the Dockerfile with relative path
image: Official Docker image from docker hub
volumes: Named volumes that keeps our data alive after restart.
network: The two services should be belong to one network.
depends_on: Dependency order, mysqldb is started before app
++***Important*** : The data base host name should be replaced by data base service name.
Ex : *jdbc:mysql://mysqldb:3306/basics?*

The project structure looks like below:



Project Structure

In here schema.sql (DDL queries) file added to create table structure and data.sql file added to load data (DML queries) while populate to spring application.

```
CREATE TABLE IF NOT EXISTS persons(
```

```
    `id` bigint(20) NOT NULL AUTO_INCREMENT,
    `name` varchar(255) DEFAULT NULL,
    PRIMARY KEY (`id`)
)ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT  INTO persons(name) VALUES('Amila');
INSERT  INTO persons(name) VALUES('Iroshan');
```

Step 4:

Run the system and inspect running containers. We can run our whole application using one docker command.

docker-compose up

You can check created docker images using : **docker images**

```
PS C:\Users\Asus\Documents\workspace-spring-tool-suite-4-4.16.1.RELEASE\basic> docker images
REPOSITORY          TAG       IMAGE ID       CREATED        SIZE
basic-api_service   latest    c2786433dde2   7 hours ago    894MB
mysql               8.0       b939d379d46e   7 days ago     514MB
```

Show Docker Images

You can check created docker containers using : **docker ps**

```
PS C:\Users\Asus\Documents\workspace-spring-tool-suite-4-4.16.1.RELEASE\basic> docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                               NAMES
e0e5cd710ef6   basic-api_service  "java -jar /home/app…"   13 seconds ago   Up 7 seconds    0.0.0.0:8080->8080/tcp              basic-api_service-1
fc71caf651d4   mysql:8.0          "docker-entrypoint.s…"   13 seconds ago   Up 9 seconds    0.0.0.0:3306->3306/tcp, 33060/tcp   basic-mysqldb-1
```

Show Docker Containers

Login in to created containers using :

api_service container = **docker exec -it basic-api_service-1 bin/sh**

```
S C:\Users\Asus> docker exec -it basic-api_service-1 bin/sh
h-4.4# ls
in  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
h-4.4# cd home/app
h-4.4# ls
om.xml  src  target
h-4.4# cd target
h-4.4# ls
lasses  generated-sources  generated-test-sources  maven-archiver  maven-status  spring_rest_docker.jar  spring_rest_docker.jar.original  test-classes
h-4.4#
S C:\Users\Asus>
```

api_service container

mysqldb container = **docker exec -it basic-mysqldb-1 bash**

```
PS C:\Users\Asus> docker exec -it basic-mysqldb-1 bash
bash-4.4# mysql -u root -p
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
bash-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.32 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| basics             |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.01 sec)

mysql> use basics;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from persons;
+----+---------+
| id | name    |
+----+---------+
|  1 | Amila   |
|  2 | Iroshan |
```

mysqldb container

Final Result

Navigate to this GET URL on you browser or any rest client.

http://localhost:8080/all

Final output

Source Code

The source code for this tutorial can be found at Github.

Hope you find this helpful. Thank you for reading this article!.

Do support our publication by following it

The Fresh Writes

We support small publishers to enhance their articles and increase their growth

medium.com

Also refer to the following articles

Threading and Multiprocessing in Python Explained

Threading and Multiprocessing are two popular methods used in Python for the parallel execution of tasks. Threading...

medium.com

Make $5000 a Month Tutoring from Home

My favorite platform. Make money from the comfort of your home.

medium.com

Nested Try Blocks In Java

In Java, a try statement can be inside the block of another try. It is called as nested try block.

medium.com

Are You Ready to Take the Plunge into the World of Blogging?

Do you have a passion for writing and creating content? Are you interested in sharing your thoughts and ideas with the...

medium.com

6 Passive Income Ideas for Making Money in 2023

Earn money while sleeping

medium.com

Spring Boot Java Docker Docker Compose MySQL

Follow

Published in The Fresh Writes

609 Followers · Last published Jan 18, 2025

We support small publishers to enhance their articles and increase their growth

Follow

Written by Amila Iroshan

249 Followers · 17 Following

Software Engineer | Open Source Contributor | Tech Enthusiast

Responses (5)

What are your thoughts? Respond

Hanni Dinh
Aug 2, 2023

Thanks bro! I finally completed all steps.

1 1 reply Reply

bridgethegap
Jun 13, 2024

Step 1 #4 and #5 could use more detail.

4: where do we empty this zip folder?

5: Create the rest api where? What file? Any chance of showing the folder strucuter?

Reply

a

amine kotni
May 17, 2024

Nice work!

Thanks !!

Reply

See all responses

More from Amila Iroshan and The Fresh Writes

In The Fresh Writes by Amila Iroshan

Create Your Own Custom Annotation In Spring Boot

Introduction

Mar 13, 2023 117 5

In The Fresh Writes by Anusha SP

Java 8 Stream API Interview Questions and Answers

Hey Reader,

Jan 15, 2023 356 4

In The Fresh Writes by Samuel Catalano

Senior Java Software Engineer Common Interview Questions

I have been living and working in the United Kingdom since 2019. I have been working with Java development since 2008 and here I will share...

Jun 6, 2024 566 5

In The Fresh Writes by Amila Iroshan

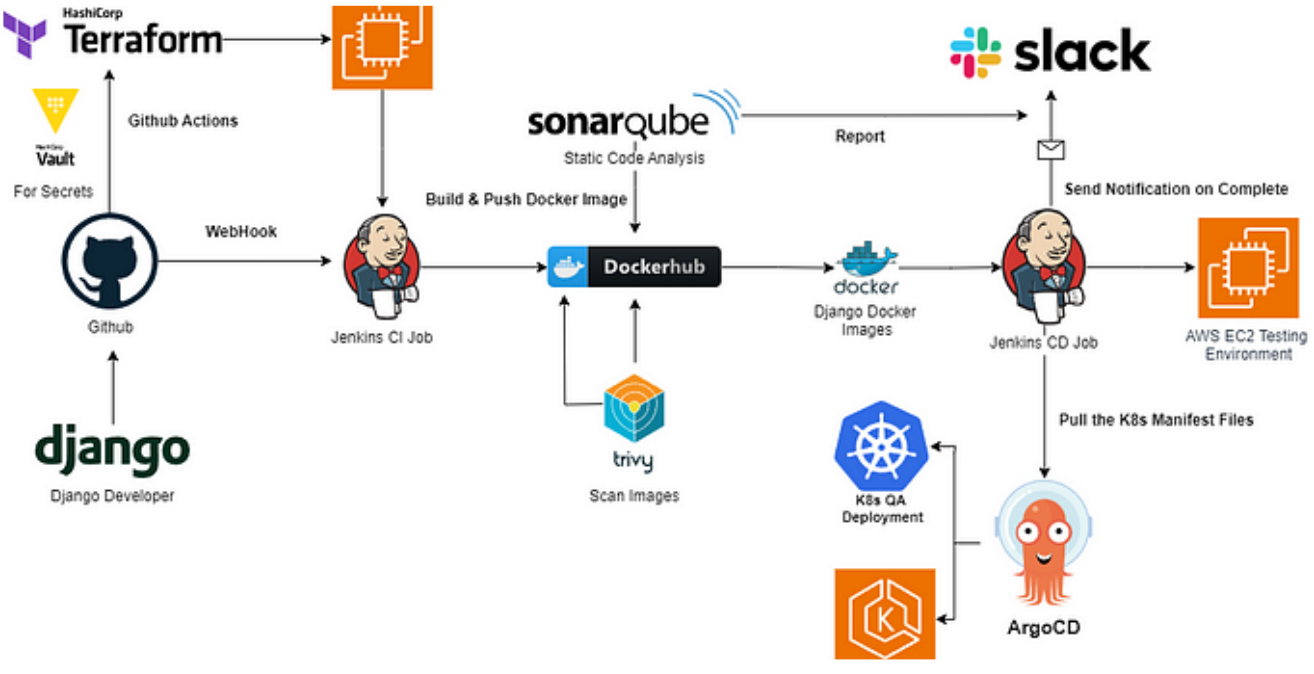
Exception Handling — Spring Boot REST API

The purpose of this article is how we can configure error/exception handling of a Spring Boot application to bring any error inside the...

Jan 27, 2024 351 4 See all from Amila Iroshan

See all from The Fresh Writes

Recommended from Medium



In Django Unleashed by Joel Wembo

Technical Guide: End-to-End CI/CD DevOps with Jenkins, Docker, Kubernetes, ArgoCD, Github Actions ...

Building an end-to-end CI/CD pipeline for Django applications using Jenkins, Docker, Kubernetes, ArgoCD, AWS EKS, AWS EC2

Apr 12, 2024 1K 21

In JavaToDev by Serxan Hamzayev

Why Financial Systems Use Stored Procedures in Spring Boot + Hibernate Applications

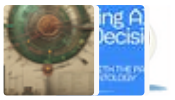
In many software projects, developers use Spring Boot and Hibernate to handle database work. However, in financial systems, there is an...

6d ago 37 3

Lists

Coding & Development
11 stories · 982 saves

General Coding Knowledge
20 stories · 1883 saves



data science and AI
40 stories · 322 saves

Natural Language Processing
1895 stories · 1556 saves

Ahmet Emre DEMİRŞEN

A Beginner’s Guide For Spring Boot, Mongo DB, Mongo Express and Rest API

Ready to take your Spring Boot and MongoDB skills to the next level? In the first part, we covered the basics of MongoDB and how to...

Aug 22, 2024 23

Vinotech

Product Management System with CRUD Operations using React.js and Spring Boot

This project is a full-stack application designed to manage products, providing CRUD (Create

Vikram Gupta

Spring Boot Profiles and How to Set It?

Create environment-specific configs in the Spring Boot Application

Sep 25, 2024

Thirupathi Pavan Sai

30 Days of Spring Boot: Day 28 — Spring Boot Advanced Topics

Master Spring Boot with Advanced Topics: Elevate Your Applications to the Next Level

Sep 6, 2024 24

See more recommendations