Docker Compose: Spring Boot and MySQL ple

ied: September 2, 2021 (https://www.bezkoder.com/docker-compose-spring-boot-mysql/)
ps://www.bezkoder.com/author/bezkoder/) Deployment
.bezkoder.com/category/deployment/), Docker (https://www.bezkoder.com/category/docker/),
//www.bezkoder.com/category/spring/)

ps://www.docker.com/) provides lightweight containers to run services in isolation frastructure so we can deliver software quickly. In this tutorial, I will show you how e Spring Boot microservice and MySQL example using Docker Compose cs.docker.com/compose/).

ts:
ot, Spring Data JPA, MySQL – Rest CRUD API example
w.bezkoder.com/spring-boot-jpa-crud-rest-api/)
ot Token based Authentication with Spring Security & JWT
w.bezkoder.com/spring-boot-jwt-authentication/)
ot + GraphQL + MySQL example (https://www.bezkoder.com/spring-boot-sql-jpa/)
ot Rest XML example – Web service with XML Response
w.bezkoder.com/spring-boot-rest-xml/)
ot: Upload CSV file data into MySQL Database (https://www.bezkoder.com/spring-d-csv-file/)
ot: Upload Excel file data into MySQL Database (https://www.bezkoder.com/spring-boot-upload-excel-file-database/)

AWS instead: Deploy Spring Boot App on AWS – Elastic Beanstalk (https://www.bezkoder.com/deploy-spring-boot-aws-eb/)

Contents [hide]

Overview

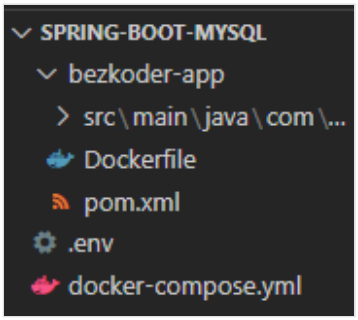
Assume that we have a Spring Boot Application working with MySQL database.
The problem is to containerize a system that requires more than one Docker container:

- Spring Boot for Rest API
- MySQL for database

Docker Compose helps us setup the system more easily and efficiently than with only Docker.
We're gonna following these steps:

- Create Spring Boot App working with MySQL database.
- Create Dockerfile for Spring Boot App.
- Write Docker Compose configurations in YAML file.
- Set Spring Boot Docker Compose Environment variables.
- Run the system.

Directory Structure:



Create Spring Boot App

You can read and get Github source code from one of following tutorials:
– Spring Boot, Spring Data JPA, MySQL – Rest CRUD API example (https://www.bezkoder.com/spring-boot-jpa-crud-rest-api/)
– Spring Boot Token based Authentication with Spring Security & JWT (https://www.bezkoder.com/spring-boot-jwt-authentication/)
– Spring Boot + GraphQL + MySQL example (https://www.bezkoder.com/spring-boot-

graphql-mysql-jpa/)
– Spring Boot Rest XML example – Web service with XML Response (https://www.bezkoder.com/spring-boot-rest-xml/)
– Spring Boot: Upload CSV file data into MySQL Database (https://www.bezkoder.com/spring-boot-upload-csv-file/)
– Spring Boot: Upload Excel file data into MySQL Database (https://www.bezkoder.com/spring-boot-upload-excel-file-database/)

Using the code base above, we put the Spring Boot project in **bezkoder-app** folder without the need of **resources**/*application.properties*. It is because Environment variables will be exported to *.env* file.

Create Dockerfile for Spring Boot App

Dockerfile defines a list of commands that Docker uses for setting up the Spring Boot application environment. So we put the file in **bezkoder-app** folder.

Because we will use Docker Compose, we won't define all the configuration commands in this Dockerfile.

bezkoder-app/*Dockerfile*

```
FROM maven:3.8.2-jdk-8

WORKDIR /bezkoder-app
COPY . .
RUN mvn clean install

CMD mvn spring-boot:run
```

Let me explain some points:

- `FROM` : install the image of the Maven – JDK version.
- `WORKDIR` : path of the working directory.
- `COPY` : copy all the files inside the project directory to the container.
- `RUN` : execute a command-line inside the container: `mvn clean install` to install the dependencies in *pom.xml*.
- `CMD` : run script `mvn spring-boot:run` after the image is built.

Write Docker Compose configurations

On the root of the project directory, we're gonna create the *docker-compose.yml* file. Follow version 3 (https://docs.docker.com/compose/compose-file/compose-file-v3/) syntax defined by Docker:

```
version: '3.8'

services:
    mysqldb:
    app:

volumes:
```

- `version` : Docker Compose file format version will be used.
- `services` : individual services in isolated containers. Our application has two services: `app` (Spring Boot) and `mysqldb` (MySQL database).
- `volumes (https://docs.docker.com/storage/volumes/)` : named volumes that keeps our data alive after restart.

Let's implement the details.

docker-compose.yml

```
version: "3.8"

services:
  mysqldb:
    image: mysql:5.7
    restart: unless-stopped
    env_file: ./.env
    environment:
      - MYSQL_ROOT_PASSWORD=$MYSQLDB_ROOT_PASSWORD
      - MYSQL_DATABASE=$MYSQLDB_DATABASE
    ports:
      - $MYSQLDB_LOCAL_PORT:$MYSQLDB_DOCKER_PORT
    volumes:
      - db:/var/lib/mysql
  app:
    depends_on:
      - mysqldb
    build: ./bezkoder-app
    restart: on-failure
    env_file: ./.env
    ports:
      - $SPRING_LOCAL_PORT:$SPRING_DOCKER_PORT
    environment:
      SPRING_APPLICATION_JSON: '{
        "spring.datasource.url"  : "jdbc:mysql://mysqldb:$MYSQLDB_DOCKER_PORT/$MYSQLD
        "spring.datasource.username" : "$MYSQLDB_USER",
        "spring.datasource.password" : "$MYSQLDB_ROOT_PASSWORD",
        "spring.jpa.properties.hibernate.dialect" : "org.hibernate.dialect.MySQL5Inno
        "spring.jpa.hibernate.ddl-auto" : "update"
      }'
    volumes:
      - .m2:/root/.m2
    stdin_open: true
    tty: true

volumes:
  db:
```

– **mysqldb**:

- `image` : official Docker image
- `restart` : configure the restart policy (https://docs.docker.com/config/containers/start-containers-automatically/#use-a-restart-policy)
- `env_file` : specify our *.env* path that we will create later
- `environment` : provide setting using environment variables
- `ports` : specify ports will be used
- `volumes` : map volume folders

– **app**:

- `depends_on` (https://docs.docker.com/compose/compose-file/compose-file-v3/#depends_on) : dependency order, **mysqldb** is started before **app**
- `build` : configuration options that are applied at build time that we defined in the *Dockerfile* with relative path
- `environment` : environmental variables that Spring Boot application uses
- `stdin_open` and `tty` : keep open the terminal after building container

You should note that the host port (`LOCAL_PORT`) and the container port (`DOCKER_PORT`) is different. Networked service-to-service communication uses the container port, and the outside uses the host port.

Docker Compose Environment variables

In the service configuration, we used environmental variables defined inside the *.env* file. Now we start writing it.

.env

```
MYSQLDB_USER=root
MYSQLDB_ROOT_PASSWORD=123456
MYSQLDB_DATABASE=bezkoder_db
MYSQLDB_LOCAL_PORT=3307
MYSQLDB_DOCKER_PORT=3306

SPRING_LOCAL_PORT=6868
SPRING_DOCKER_PORT=8080
```

Run the Spring Boot microservice with Docker Compose

We can easily run the whole with only a single command:

```
docker-compose up
```

Docker will pull the MySQL and Maven images (if our machine does not have it before).

The services can be run on the background with command:

```
docker-compose up -d
```

```
$ docker-compose up -d
Creating network "spring-boot-mysql_default" with the default driver
Creating volume "spring-boot-mysql_db" with default driver
Pulling mysqldb (mysql:5.7)...
5.7: Pulling from library/mysql
e1acddbe380c: Pull complete
bed879327370: Pull complete
03285f80bafd: Pull complete
ccc17412a00a: Pull complete
1f556ecc09d1: Pull complete
adc5528e468d: Pull complete
1afc286d5d53: Pull complete
4d2d9261e3ad: Pull complete
ac609d7b31f8: Pull complete
53ee1339bc3a: Pull complete
b0c0a831a707: Pull complete
Digest: sha256:7cf2e7d7ff876f93c8601406a5aa17484e6623875e64e7acc71432ad8e0a3d7e
Status: Downloaded newer image for mysql:5.7
Building app
Sending build context to Docker daemon  22.02kB
Step 1/5 : FROM maven:3.8.2-jdk-8
 ---> 80704b8c5fbd
Step 2/5 : WORKDIR /bezkoder-app
 ---> Running in f63e76f45fcc
Removing intermediate container f63e76f45fcc
 ---> 10802ac64cea
Step 3/5 : COPY . .
 ---> 9dcd16082f00
Step 4/5 : RUN mvn clean install
 ---> Running in 288bea890f74
[INFO] Scanning for projects...
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/bo
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boo
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/bo
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boo
...
[INFO] Installing /bezkoder-app/target/spring-boot-data-jpa-0.0.1-SNAPSHOT.jar to /ro
[INFO] Installing /bezkoder-app/pom.xml to /root/.m2/repository/com/bezkoder/spring-b
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  02:41 min
[INFO] Finished at: 2021-08-18T04:10:08Z
[INFO] ------------------------------------------------------------------------
Removing intermediate container 288bea890f74
 ---> adddf4648410
Step 5/5 : CMD mvn spring-boot:run
 ---> Running in c81f8028e2eb
Removing intermediate container c81f8028e2eb
 ---> 1f710daedbf2
```

```
Successfully built 1f710daedbf2
Successfully tagged spring-boot-mysql_app:latest
WARNING: Image for service app was built because it did not already exist. To rebuild
Creating spring-boot-mysql_mysqldb_1 ... done
Creating spring-boot-mysql_app_1     ... done
```

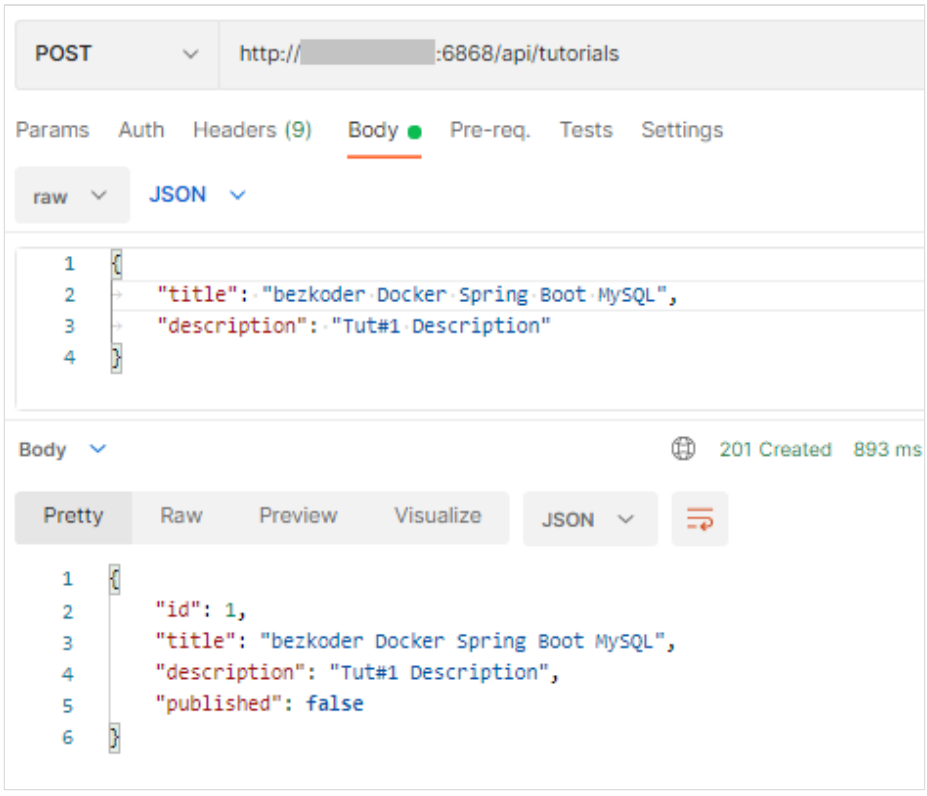
Now you can check the current working containers:

```
$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATU
5ad28f104e8b   spring-boot-mysql_app  "/usr/local/bin/mvn-…"   3 minutes ago    Up 3
ba9281773e7f   mysql:5.7              "docker-entrypoint.s…"   3 minutes ago    Up 3
```

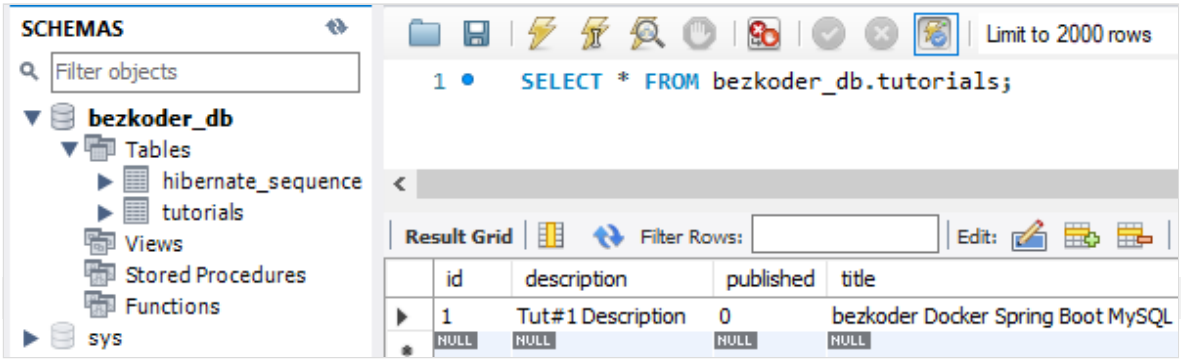
And Docker images:

```
$ docker images
REPOSITORY              TAG             IMAGE ID       CREATED          SIZE
spring-boot-mysql_app   latest          1f710daedbf2   5 minutes ago    672MB
mysql                   5.7             6c20ffa54f86   6 minutes ago    448MB
maven                   3.8.2-jdk-8     80704b8c5fbd   6 minutes ago    525MB
```

Send a HTTP request to the Spring Boot – MySQL system:



Check MySQL Database:



Stop the Application

Stopping all the running containers is also simple with a single command:

```
docker-compose down
```

```
$ docker-compose down
Stopping spring-boot-mysql_app_1     ... done
Stopping spring-boot-mysql_mysqldb_1 ... done
Removing spring-boot-mysql_app_1     ... done
Removing spring-boot-mysql_mysqldb_1 ... done
Removing network spring-boot-mysql_default
```

If you need to stop and remove all containers, networks, and all images used by any service in *docker-compose.yml* file, use the command:

```
docker-compose down --rmi all
```

Conclusion

Today we've successfully created Docker Compose file for Spring Boot application and MySQL. Now we can connect Spring Boot to MySQL with Docker on a very simple way: *docker-compose.yml*.

You can apply this way to one of following project:
– Spring Boot, Spring Data JPA, MySQL – Rest CRUD API example (https://www.bezkoder.com/spring-boot-jpa-crud-rest-api/)
– Spring Boot Token based Authentication with Spring Security & JWT (https://www.bezkoder.com/spring-boot-jwt-authentication/)
– Spring Boot + GraphQL + MySQL example (https://www.bezkoder.com/spring-boot-graphql-mysql-jpa/)
– Spring Boot Rest XML example – Web service with XML Response (https://www.bezkoder.com/spring-boot-rest-xml/)
– Spring Boot: Upload CSV file data into MySQL Database (https://www.bezkoder.com/spring-boot-upload-csv-file/)
– Spring Boot: Upload Excel file data into MySQL Database (https://www.bezkoder.com/spring-boot-upload-excel-file-database/)

If you want to deploy the system on AWS, please visit:
Deploy Spring Boot App on AWS – Elastic Beanstalk (https://www.bezkoder.com/deploy-spring-boot-aws-eb/)

Happy Learning! See you again.

Source Code

The source code for this tutorial can be found at Github (https://github.com/bezkoder/docker-compose-spring-boot-mysql).

You can deploy the container on Digital Ocean (https://www.digitalocean.com/?refcode=560b7a03275b&utm_campaign=Referral_Invite&utm_medium=Referral_Program&utm_source=CopyPaste) with very small budget: **5$**/month.
Using referral link below, you will have **200$** in credit over **60** days. After that, you can stop the VPS with no cost.

(https://www.digitalocean.com/?refcode=560b7a03275b&utm_campaign=Referral_Invite&utm_medium=Referral_Program&utm_source=badge)

deploy (https://www.bezkoder.com/tag/deploy/) deployment (https://www.bezkoder.com/tag/deployment/)

docker (https://www.bezkoder.com/tag/docker/)

docker compose (https://www.bezkoder.com/tag/docker-compose/)

dockerize (https://www.bezkoder.com/tag/dockerize/) mysql (https://www.bezkoder.com/tag/mysql/)

rest api (https://www.bezkoder.com/tag/rest-api/) spring boot (https://www.bezkoder.com/tag/spring-boot/)

6 thoughts to “Docker Compose: Spring Boot and MySQL example”

luis ortiz

February 22, 2022 at 5:18 pm (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-19689)

Hi,

Thanks for the tutorial, I have a questio. Why allways get this message:

curl: (52) empty reply from server -> From curl command cmd
Error: socket hang up -> From postman petition

Thanks

Adrian

January 9, 2022 at 5:50 pm (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-18038)

Do I have to rebuild the image everytime I make a change in the spring boot app?

Barrrettt

December 24, 2021 at 12:32 pm (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-17135)

omg, thanks!
is the healthcheck not necessary in the mysqldb service?

Josh

November 23, 2021 at 2:16 pm (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-15588)

Hi, thanks for the tutorial, its great! Could you explain how the tables within the database are being created? I am currently also looking at the tutorial you made for implementing jwt tokens on a java backend and wanted to be able to create the roles, users etc tables that you had. Once again, thanks for all the great stuff!

jiji
November 11, 2021 at 4:29 pm (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-15128)

hello, how about the front-end part using angular ?

bezkoder
November 12, 2021 at 3:05 am (https://www.bezkoder.com/docker-compose-spring-boot-mysql/#comment-15147)

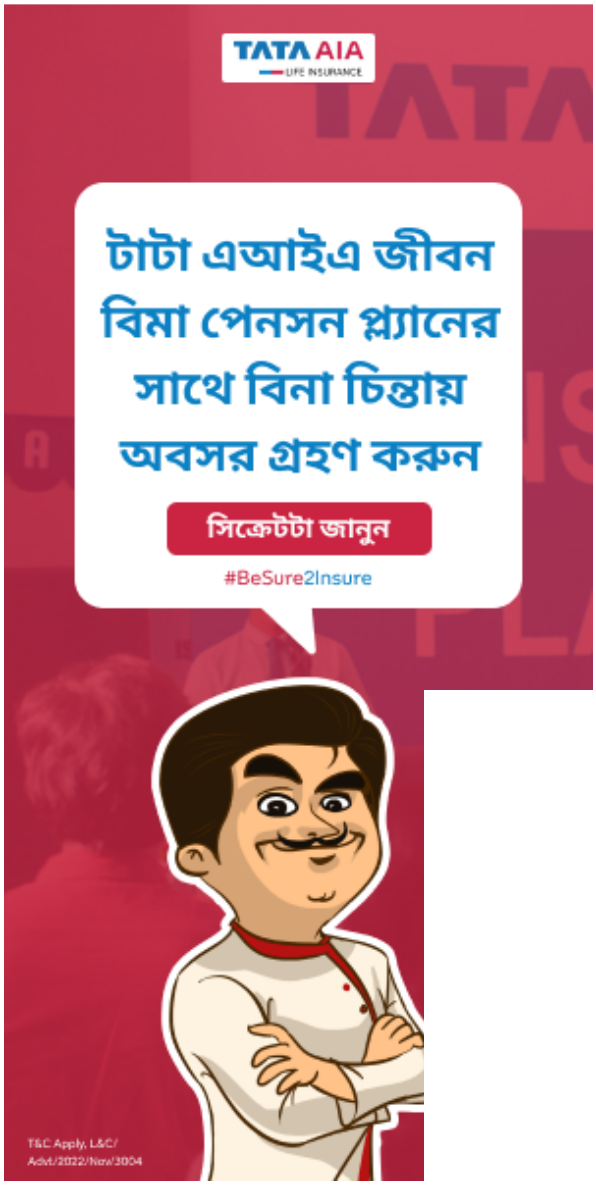
Hi, I will write the tutorial when having time 🙂

Comments are closed to reduce spam. If you have any question, please send me an email.

‹ Spring Boot + React Redux example: Build a CRUD App (https://www.bezkoder.com/spring-boot-react-redux-example/)

Docker Compose: Node.js Express and MongoDB example › (https://www.bezkoder.com/docker-compose-nodejs-mongodb/)

Search...


TATA AIA
LIFE INSURANCE
টাটা এআইএ জীবন
বিমা পেনসন প্ল্যানের
সাথে বিনা চিন্তায়
অবসর গ্রহণ করুন
সিক্রেটটা জানুন
#BeSure2Insure
T&C Apply, L&C/
Advt/2022/Nov/3004

FOLLOW US

(https://www.youtube.com/channel/UCp0mx9RH0JxaFsmMvKXA86Q)

(https://facebook.com/bezkoder)

(https://github.com/bezkoder)

TOOLS

Json Formatter (https://www.bezkoder.com/json-formatter/)

. Privacy Policy (https://www.bezkoder.com/privacy-policy/)

Contact (https://www.bezkoder.com/contact-us/) About (https://www.bezkoder.com/about/)