Title: Spring Docker container cannot access Postgres Docker container
Post Body:

The Dockerfile of my spring-boot app:

```
FROM openjdk:8-jdk-alpine VOLUME /tmp COPY target/media-0.0.1-SNAPSHOT.jar app.jar ENTRYPOINT ['java', '-jar', '/app.jar']
```

`application.yml`

```
spring:   datasource:     url: jdbc:postgresql://localhost:5432/media     username: postgres     password: postgres     hikari
```

and here is the `docker-compose.yml`:

```
version: '3' services:   db:     image: postgres     ports:       - '5432:5432'     environment:       POSTGRES_DB: media
```

Running `docker-compose up --build` results in:

> app_1 | org.postgresql.util.PSQLException: Connection to 0.0.0.0:5432 refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections. app_1

My guess is that the spring app tries to connect to postgres before postgres is ready, but I get the following log:

> db_1 | 2019-05-18 19:05:53.692 UTC [1] LOG: database system is ready to accept connections

Accepted Answer:

The main purpose of Docker Compose is to spin up a set of Docker containers, which will then function as independent entities. By default, all containers will have a virtual network connection to all others, though you can change that if you wish; you will get that feature, since you have not specified a custom configuration.

Each of the containers will get a virtual IP address inside the virtual network set up by Docker. Since these are dynamic, Docker Compose makes it easier for you by creating internal DNS entries corresponding to each service. So, you will have two containers, which can be addressed as `app` and `db` respectively, either from themselves or the other. If you have ping installed, you can ping these names too, either via `docker-compose exec`, or via a manually-created shell.

Thus, as we discovered in the comments, you can connect from `app` to `jdbc:postgresql://db:5432/media`, and it should work.

Highest Rated Answer: None