Title: RabbitMq Spring Boot deploy docker

Post Body:

I've got a spring boot application with producer and receiver which uses the local rabbitmq to send and receive messages. I am trying to deploy both the app and rabbit on docker container and once deployed run the boot app. Once the app is started it has a producer which implements Spring CommandLineRunner meaning that the app immediately starts sending messages so there must be a rabbitmq broker running prior. I am very new to docker. I tried so far:

1). I created image for the spring boot app as the following

```
FROM java:8 EXPOSE 8080 ADD /target/MessagingApp.jar MessagingApp.jar ENTRYPOINT ['java','-jar','MessagingApp.jar']
```

2). Then created docker.compose file for rabbit and my newly created image

```
rabbitmq:   image: rabbitmq:management   ports:     - '5672:5672'     - '15672:15672' messagingapp:   image: messagingapp:late
```

I then ran docker-compose up and I can see rabbit started to some extend and then spring boot app but fails sending messages with

```
at com.rabbitmq.client.impl.SocketFrameHandlerFactory.create..
```

I think my issue is that my spring boot app automatically tries to connect to the localhost rabbit host. So how do I make it point to the docker rabbitmq server?

Any help?

Accepted Answer:

Try update part of `links` for `depends_on`. Your application probably start before `messagingapp`.

Part of documentation for `depends_on`

> docker-compose up will start services in dependency order. When docker-compose execute V2 files, it will automatically build a network between all of the containers defined in the file, and every container will be immediately able to refer to the others just using the names defined in the docker-compose.yml file.

But

> Note: depends_on will not wait for db and redis to be “ready” before starting web - only until they have been started.

For that check Controlling startup order.

You need add `command` for checking state of service. More in documentation...

```
depends_on:       - 'db'     command: ['./wait-for-it.sh', 'db:5432', '--', 'python', 'app.py']
```

Highest Rated Answer:

Your Dockerfile and docker-compose.yml files seem correct except for two things:

> instead of the keyword `links`, you need to use `depends_on`. This will build and run RabbitMQ instance first, then your web service.
>
> you should add this line under your messagingapp service:

-->

```
environment: -  SPRING_RABBITMQ_HOST=rabbitmq
```

This will let your Spring Boot application to recognize the RabbitMQ image in Docker.

So, try to update your docker-compose.yml file like this:

```
rabbitmq:   image: rabbitmq:management   ports:     - "5672:5672"     - "15672:15672" messagingapp:   image: messagingapp:late
```