Title: How to create containerized Spring Boot App on Raspberry Pi 4?

Post Body:

What am I wanting to do?

I want to convert to run my ***containerized of Spring-Boot's app*** which is can run on amd64 systems, on Raspberry Pi 4's Docker. How can I fix the my problem?

What's the running OS on Raspberry Pi?

I've installed ***Ubuntu Server 20.04.2 LTS for arm64 architectures*** using via ***Raspberry Pi Imager v1.6.1*** on ***Raspberry Pi 4***.

What are the steps I have done to fix the problem?(problem is still continue)

In default version does not work on my ***Raspberry Pi.*** Thus, I've tried some couple of things at below.

I changed row of "FROM" from ***openjdk:15-jdk-slim*** to ***arm64v8/openjdk:17*** at my ***dockerfile*** as you can see:

```
FROM arm64v8/openjdk:17 COPY . /projects/red-dir WORKDIR /projects/red-dir ARG JAR_FILE=target/*.jar COPY ${JAR_FILE} /project
```

Afterwards, I've built and pushed my app using with ***Maven*** and ***Docker***:

```
$ ./mvnw clean package -Dmaven.test.skip=true && java -jar target/my_red_app.jar $ docker build -t user/my_red_app:v1.0.0 .
```

Then, My app's depend on a MongoDB, so I changed this ***image*** from ***mongo*** to ***arm64v8/mongo*** on ***docker-compose.yml*** as you can see at below:

```
version: "3" services:     mongodb:         container_name: mongodb         image: arm64v8/mongo         restart: always
```

Things that work and things that don't work

MongoDB is running. I can reach using via ***MongoDBCompass***, but my ***Spring-Boot app*** doesn't work. In `docker-compose up` command, I've encountered an output like this:

```
... ... colour_app    | standard_init_linux.go:219: exec user process caused: exec format error colour_app    | standard_init_
```

Let me know what things I didn't catch. Thank you for reading.

Accepted Answer:

I've solved my problem and tried to create simple guide for ones who are encountered this problem.

BUILD FOR OTHER OS (multi-architecture / buildx command)

1. Turn On "experimental" feature's on Docker Settings.

   For Linux OS:

   1. `sudo nano /etc/docker/daemon.json`
   2. Add below content to it:

      ```
      {   "experimental": true }
      ```

   For Win10 OS / MAC OS:

   1. Open **Docker Desktop** application.
   2. Go to Settings.
   3. Select ***Docker Engine*** Tab on the left-side.
   4. Find `"experimental": false`.
   5. Change it to `"experimental": true`.
2. Restart Docker.

   Create multi-architecture os. (***Important****: The* ***image*** *you are using should contain the* ***OS*** *you want to create it*)

   Creating a just **SINGLE OS** :

   1. You can use load or push commands.

      `--load` means that docker saves it to local disk:

```
docker buildx build --load --platform linux/arm64 -t <dockerhub_username>/<repository_name>:<tag_name> .
```

`--push` means that docker doesn't save it to local disk but save it to cloud (docker hub):

```
docker buildx build --push --platform linux/arm64 -t <dockerhub_username>/<repository_name>:<tag_name> .
```

- `--load` and `--push` can not be set together.

Creating a **MANY OS** :

You can just use push command with many OS.

`--push` means that docker doesn't save it to local disk but save it to cloud (docker hub):

```
docker buildx build --push --platform linux/amd64,linux/arm64,linux/ppc64le -t <dockerhub_username>/<repository_name
```

Highest Rated Answer: None