Title: Docker unable to access jar file

Post Body:

The docker container is not able to access the jar file, that is being accessed over the mount point `/my/project/dir`. I am certain it is not a permission issue, because I changed the access rights locally, so it should be able to read/write/execute it.

This is the `Dockerfile`:

```
FROM tomcat:9-jre8 RUN apt-get update && apt-get install librrds-perl rrdtool -y VOLUME ['/data/rrdtool', '/my/project/dir'] E
```

And this is the `docker-compose.yml` file:

```
version: '2' services: db:    container_name: db1    image: mysql:8    restart: always    environment:        MYSQL_ROOT_PASSWO
```

This is the output from `docker-compose logs spring`:

```
Error: Unable to access jarfile /my/project/dir/build/libs/spring-project-0.1.0.jar
```

Accepted Answer:

I don't see you copying the `jar` into the container anywhere. You should try moving a `VOLUME` declaration from `Dockerfile` to the `compose` file into the `spring` service like this:

```
  volumes:      - /my/project/dir:/app
```

And then inside Dockerfile you should point to the dir:

```
ENTRYPOINT [ 'java','-jar','/app/build/libs/spring-project-0.1.0.jar' ]
```

Later on if you'd like to deploy it (for example) you should copy the project files directly into the image instead of utilizing the `volumes` approach. So in `Dockerfile` you'd then do:

```
COPY . /app
```

instead of `VOLUME [..]`

Putting it all together:

development:

Dockerfile:

```
FROM tomcat:9-jre8 RUN apt-get update && apt-get install librrds-perl rrdtool -y ENTRYPOINT [ 'java','-jar','/app/build/libs/s
```

compose-file:

```
version: '2' services:     [..]     spring:        container_name: spring-boot-project        build: .        links:
```

deployment:

Dockerfile (that is placed inside project's folder, docker build requires it's build context to be in a current directory):

```
FROM tomcat:9-jre8 RUN apt-get update && apt-get install librrds-perl rrdtool -y COPY . /app ENTRYPOINT [ 'java','-jar','/app/
```

compose-file:

```
version: '2' services:     [..]     spring:        container_name: spring-boot-project        build: .        links:
```

Highest Rated Answer:

If you are using `Spring-Boot` Project with `Maven` build. Try with below

Dockerfile.

```
FROM maven:3.8.4-openjdk-17 as maven-builder COPY src /app/src COPY pom.xml /app  RUN mvn -f /app/pom.xml clean package -Dskip
```

```
   dockube-spring-boot.jar // replace with your generated jar name
```

Here is the [Sample Code Available](#)