Title: How to use maven local repository in Multi-stage docker build?

Post Body:

I've been trying to create a multi-stage docker build for my spring-boot-application. The problem is on every change on the master branch, the pom.xml file changes (thanks to maven release plugin), so it's kind of hard to make use of docker build-cache during the build stage and all the dependencies will be downloaded every time I run the build. As far as I know, I can't mount volumes during the building of the image. The experimental version of the docker allows you to mount volumes during the build but it's still experimental so I'm trying to avoid it.

So I decided to try my luck with docker-compose, where the first service will run 'mvn package' inside the container and create a jar file inside a shared volume, which will be then used by the second service to build its own image (basically adding that jar inside its image), which will be published to docker hub. Whatever I was trying to do didn't seem right. Now I'm not really expert of docker-compose here, but is this the right approach to do it?
I'm using `Docker version 18.09.0, build 4d60db4`

```
version: '3' #Specify all the services you want to create services:   #service name   build-service:     context: .     docker
```

Is there any way to make use of local .m2 repository during the first stage. Or any other suggestions are welcome.

Accepted Answer: None

Highest Rated Answer:

The short answer is that you cannot. The only way you could use the maven repository during build would be to copy it inside the image in the first stage. But you cannot do that because normally the location of the maven repository is outside your build context. Of course you can change that for your project (place the .m2 folder in your current project) and then this approach might work. I am not recommending this approach, I am merely mentioning it as an option.

However, I think you can solve your issue by following the best practice of not using Docker during development. While Docker is an awesome tool, it does slow down development. The build and push of the image should be delegated to your CI/CD pipeline (Jenkins, Gitlab CI, etc.). During day to day activities it is better and faster to just run your maven builds locally.