Title: Use docker compose with compiling
Post Body:

I want to deploy a maven application with docker container and if possible also test with docker, but a have some problems.

I because of using java I need to compile my application before using is.

In the process of compiling there also running unit test, which need a database connection.

For testing I used a database container started from hand who run on localhost:5432.

If I start docker-compose now this causes an error because the container can't reach localhost:5432 any more. If I write postgres:5432 in my application.properties it does not compile because of the unknown host postgres.

How to handle this. Is there a way to start a with maven and an with postgres to building time.

As you see I am new to docker-compose, and don't have a workflow yet.

Thanks for your help

Accepted Answer:

You should use your existing desktop-oriented build process to build and test the application and only use Docker to build the final deployment artifact. If you are hard-coding the database location in your source code, there is lurking trouble there of exactly the sort you describe (what will you do if you have separate staging and production databases hosted by your cloud provider?) and you should make that configurable.

During the `docker build` phase there's no way to guarantee that any particular network environment, external services, or DNS names will be present, so you can't do things like run integration tests that depend on an external database. Fortunately that's a problem the software engineering community has spent a long time addressing in the decades before Docker existed. While many Docker setup are very enthusiastic about mounting application source code directly into containers, that's much less useful for compiled languages and not really appropriate for controlled production deployments.

In short: run Maven the same way you did before you had Docker, and then just have your Dockerfile COPY the resulting (fully-tested) `.jar` file into the image.

Highest Rated Answer: None