# Starting a local server (on MacOS): Existing Docker container

NOTE: This procedure assumes that mySQL is running locally. See below for instructions on how to do this.

**1. Make sure the Docker app is launched on MacOS.**

**2. Run the server: Do this from MacOS**

**a) Open a terminal window**

**b) Then,**

cd ~/Desktop/Apps/SyncServerII/SyncServerII

**c) Last—run the server as a docker image**

./devops/runLocally.sh ~/Desktop/Apps/SyncServerII/Private/Server/ClientTesting-local.json latest

**3. Get the logs**

**a) Open a new terminal window**

**b) Change into the directory**

cd ~/Desktop/Apps/SyncServer.Run

**c) tail the log**

tail -f output.log

**4) Test the server:**

From a browser:

<http://localhost:8080/HealthCheck/>

(trailing slash is important)

# Building server

**1. Make sure the Docker app is launched on MacOS.**

**2. Start Docker build container**

# docker run --rm -i -t -v /Users/chris/Desktop/NewSyncServer/:/root/Apps crspybits/swift-ubuntu:5.2.3

**3. In the container**

cd root/Apps/ServerMain

./Tools/clean.sh

./Tools/build.sh

If build.sh has problems (e.g., is very slow to build, and perhaps stalls/blocks), try:

./Tools/build.sh verbose

However, that’s not what helped in my most recent go-around with this (5/30/20). Rather, after floundering for several hours, I updated from Swift 5.0.1 in my Docker build container (devops/Docker/Building) to Swift 5.2.3 and now my

./Tools/build.sh

works with no delays. I assume there have been dependency resolution changes in the last few Swift releases.

I haven’t seen many other references to these issues online. Here’s one:

<https://stackoverflow.com/questions/47431510/swift-package-manager-not-resolving>

# Starting server from local build

**Starting mySQL for local running of the server**

**Checking the local database**

**Creating** ClientTesting-local.json for running the server.