Building, Running, & Packaging Dockerfiles

This page is on what to do after you've created your docker file and how to run and package it to be uploaded.

> This page assumes your knowledge and understanding of Dockerfiles, how to create them properly, and general command line knowledge.

Building Docker Files

To build a docker file you need to utilize docker build commands. Before anything you have to make sure you're within the directory of your docker file. To do this simply right-click on the folder containing your docker file and "open in terminal" or in a command terminal cd into the directory.

Build Commands

To build a container image using the Dockerfile example from the previous section, you can use the `docker build` command:

```
sudo docker build -t test .
```

This is an example of the docker build command in Linux requiring the "sudo" function, if you're on Windows/Mac you might not need it. The `-t test` option specifies the name of the image.

The single dot (`.`) at the end of the command sets the build context to the current directory. This means that the build expects to find the Dockerfile and the `tm-voicebox-v.3.7.4.py` file in the directory where the command is invoked. If those files aren't there, the build fails.

There are also custom-build commands used for Tape Measure but we'll get to that later on. If you want to explore every build command option outside of this basic example go here.

Running Docker Containers

Once you successfully build your docker container you need to access or run it. This is where the run command comes in. Again, you'll have to be within the directory of your docker file for all of the

following directions.

Run Commands

To run the built container from the previous example you can use the `docker run` command.

```
sudo docker run -it test
```

In this instance "test" is the name of the container and, again, sudo might be required if you're on Linux. If you want to explore everything docker run has to offer outside of this basic example go here.

Packaging Docker Containers

To package a docker container and upload it into TM we need the docker image container to be compressed into a physical file instead of a container image. In order to do this we need to compile the docker container into a .tgz file. Below are the instructions to do so for Tape Measure.

Dockerfile to .tgz

The following terminal code segments were run in a Linux system, if you're on Windows/Mac you might need to adjust some of the terminal commands, if you run into a problem message @Cai or @dwight_2 on the discord server.

```
sudo docker run --rm --privileged multiarch/qemu-user-static --reset -p yes
```

This command runs a throwaway container with full privileges, ensuring it has the latest qemu emulation binaries via the multiarch/qemu-user-static image. This is often used to run containers designed for another architecture (like ARM) on an x86 host which is what TM is built on.

```
sudo docker build -t tm-voicebox-v.10.6.8 --platform linux/arm64 --build-arg
BOSDYN_CLIENT_USERNAME=admin --build-arg BOSDYN_CLIENT_PASSWORD=<Insert Password> .
```

This command builds a Docker image for arm64 called tm-voicebox-v.10.6.8, passing credentials via build arguments. The Dockerfile to build the image is expected to be in the current working directory. To get the password please contact @dwight_2 on the RCCF discord server.

```
sudo docker save tm-voicebox-v.10.6.8 > tm-voicebox-v.10.6.8.tgz
```

This command saves the Docker image to a portable compressed archive file for transfer or backup. The tar archive contains all the layers and metadata of the image. The archive file can then be loaded into Docker again using `docker load` on TM but more on that on the next page.

Revision #9
Created 18 November 2023 04:31:35 by Caicheng Li
Updated 24 January 2024 00:50:01 by Caicheng Li