Guide to Docker

All TAs, but especially Gibran and Prindle

August 18, 2020

Contents

1 Installing Docker

PLEASE READ THIS SECTION CAREFULLY. The installation process is easy, but there are multiple versions of Docker and you have to be careful to install the right one.

1.1 macOS

If you are using a Mac, then Docker installation is super easy. As long as your Mac is a 2010 model or newer and is running macOS El Capitan 10.11 or newer, you can head over to Docker's website at the following link and follow the instructions to install the Community Edition of Docker: `https://docs.docker.com/docker-for-mac/install/`

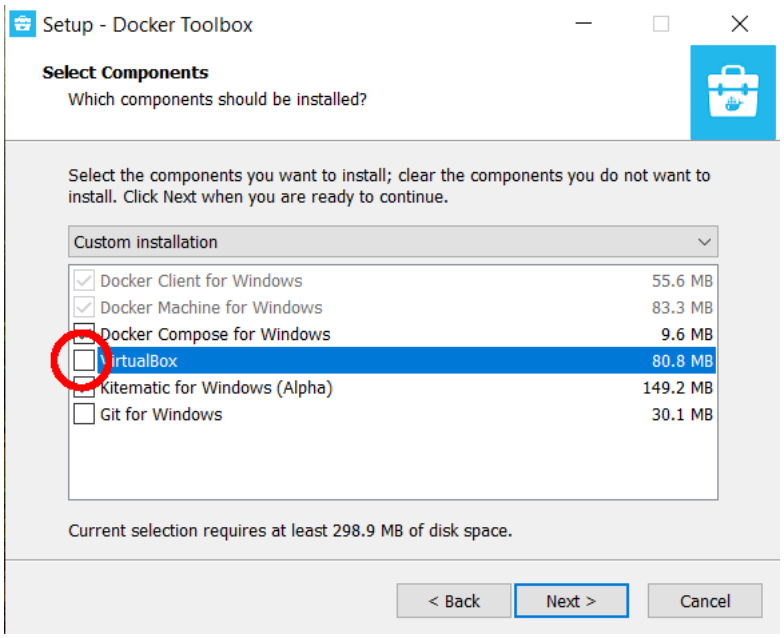
If your Mac is older than 2010, come to office hours or ask on Piazza and we can discuss a solution.

1.2 Windows

First, install the latest version of VirtualBox, which can be found at the following link. `https://www.virtualbox.org/wiki/Downloads`, Click on `Windows Host` to download, then run the installer.

Second, install Docker Toolbox. You can find instructions here: `https://docs.docker.com/toolbox/toolbox_install_windows/`

IMPORTANT: When installing Docker Toolbox, make sure to uncheck VirtualBox, as shown in the following picture:



DO NOT INSTALL DOCKER FOR WINDOWS. DISREGARD THE LINKED PAGE IF IT TELLS YOU TO. USE DOCKER TOOLBOX!

1.3 Linux

In order to install Docker on Linux (specifically Ubuntu), you need to be running at least Ubuntu 16.04. If you are running a different flavor of Linux, you should be able to find instructions online on how to install it. If you are running Ubuntu 16.04 or newer, follow the instructions here: `https://docs.docker.com/install/linux/docker-ce/ubuntu/`

2 CS 2110 Docker Image

We have created a Docker image for the students of CS 2110 that contains all of the tools that they will need throughout this course. The image itself is hosted on Docker Hub (https://hub.docker.com/r/gtcs2110/cs2110docker/), but we have developed a script for you to use so you don't need to worry about that. The image is setup to be a very basic Ubuntu 18.04 machine that includes the following tools that you need for this class:

- CircuitSim (for circuits)
- complx (for assembly)
- gcc/gdb (for C)

One thing that it does not include is a text editor. **STUDENTS SHOULD NOT EDIT ANY FILES INSIDE OF THE CONTAINER!**

3 Starting up the Docker Image

Once you have installed Docker, you are ready to download and run the CS 2110 Docker image. To do that, download the **cs2110docker.sh** script from `Files` → `Docker Resources` on Canvas, and save it in the directory that you plan on putting all of your code for this class in. Next, we need to make sure Docker is up and running so we can run the script.

3.1 macOS

If Docker is running, you will see a little whale on your menubar in the top right corner of your screen. If it isn't, go into your Applications folder and double click on the Docker icon to launch it. Click on the whale in the menubar and wait for the status to say "Docker is running." Once its running, you need to give proper permissions to the the cs2110docker script. Open a terminal window, `cd` into the directory where you put the script, and run the following:

```
sudo chmod +x cs2110docker.sh
```

Once you've set the permissions (which you only have to do on the first run), you can run the script using the following command:

```
./cs2110docker.sh
```

This may take a moment if it is your first time running the script, but if everything goes as planned, you should see eventually see the message "`Successfully launched CS 2110 Docker container.`"

3.2 Windows

The installation process put an icon on your desktop for Docker Quickstart:

Double click on this icon to launch the Docker terminal. From this terminal window, cd into the directory where you put the cs2110docker.sh script. Once you're there, run it using the following command:

```
./cs2110docker.sh
```

IMPORTANT: On Windows, the `cs2110docker.sh` file must be under `C:/Users`

This may take a moment if it is your first time running the script, but if everything goes as planned, you should see eventually see the message "`Successfully launched CS 2110 Docker container.`"

3.3 Linux

If you are running Linux, the installation process should have set the Docker daemon to start on boot, so you don't need to worry about launching it manually. Before you can run the script, you have to set up the user group for Docker. Open a terminal window and run the following commands:

```
sudo groupadd docker
sudo usermod -aG docker $USER
```

Once you've created the user group (which you only have to do on the first run), you can run the script using the following command:

```
./cs2110docker.sh
```

This may take a moment if it is your first time running the script, but if everything goes as planned, you should see eventually see the message "`Successfully launched CS 2110 Docker container.`"

4 Using the CS 2110 Docker Container

There are two ways to run the Docker image—using a browser or using a VNC client. When you ran the cs2110docker.sh script, the last line outputted a URL, which should be something along the lines of

```
http://<ip-address>:6901/vnc.html
```

This URL gives you access to the container from a web browser, which is the easiest way to use it. Open up your browser and go to the URL. You should see this page:

Click connect and enter the super secret password: **cs2110rocks**

If you don't want to use the browser, you can also connect using any VNC client. In order to connect, use the same base URL as the browser, but use the port **5901** with nothing after it.

That's it! You should be up and running with what looks like a virtual machine running in your browser! If you open up the File Manager and look along the left, you should see a device called **host**. This is the folder that you had the *cs2110docker.sh* script in. It has been mounted inside of the container. **When you are doing your work, you should edit all of you files in this folder ON YOUR HOST MACHINE. Then, use the Docker container to run the code. DO NOT EDIT FILES INSIDE THE CONTAINER. THEY WILL DISAPPEAR. PLEASE DON'T DO IT. PLEASE.** You will not get extensions on projects because you saved stuff where you should not have. We are making this very clear up front. Don't do it.

When you are done using the container, you can shut it down by running

```
./cs2110docker.sh stop
```

If you are interested in running the container in the shell (no graphics), you can run

```
./cs2110docker.sh -it
```

5 Wrap up

That's about all that there is to the CS 2110 Docker container setup. If you have any questions or issues, please post on Piazza **under the Docker folder**, so we can sort through Docker questions efficiently.

6 Addendum: What even is Docker??

I'm so glad you asked. In order to answer this question we have to go back in time.

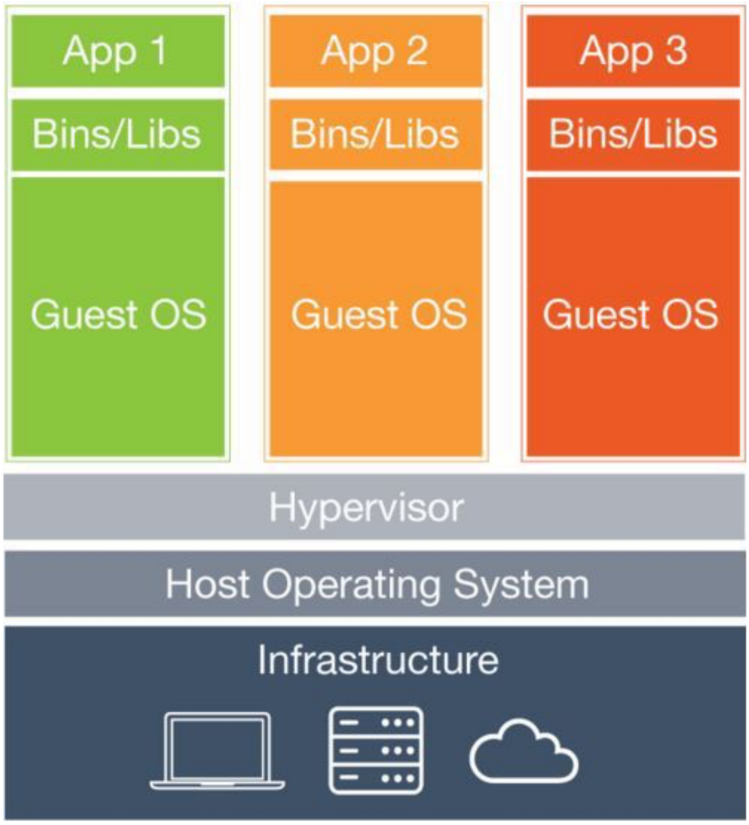
6.1 Physical Servers

Back in the stone age, when application developers wanted to deploy their application, they first had to procure a server. These servers were very expensive, and for various reasons they could run a very small

number of applications on each server. They also had to make sure their servers were powerful enough to handle a high load, so at times of lower load, there were a lot of wasted resources that were not doing anything. Monsters stalked the night, storms filled the skies, pestilence ran through the streets, and death was common. This obviously was a pretty big problem.

6.2 Virtual Machines

In order to help minimize the amount of resources that were wasted on servers, people started using "Virtual Machines." Virtual machines are essentially computers within computers. First, you start with a physical server and install an operating system. On top of this operating system, you install something called a "hypervisor." A hypervisor is a type of virtualization technology that allows us to emulate the hardware of a computer and install more operating systems on top of the main operating system. These guest operating systems are called virtual machines. With virtual machines, we can have one physical server and run multiple operating systems, which means that we can run multiple applications and thus not waste a ton of space. Problem solved! Here is a picture to illustrate how virtual machines work:



6.3 What now?

So we have fixed the problem of wasted resources. Time to kick back and put our feet up and be happy with everything we've accomplished, right? Well, if you take a look at the picture, guest operating systems are duplicated multiple times and taking up a bunch of space. Creating entire guest operating systems is incredibly wasteful. If you think about it, there is a ton of stuff that is duplicated. Each guest operating system has copies of the same base files and programs that make it up. In addition, emulating multiple operating systems takes a TON of system resources, which does not leave much for the applications themselves. So while we have solved the problem of having extra resources that are not utilized, we now have the

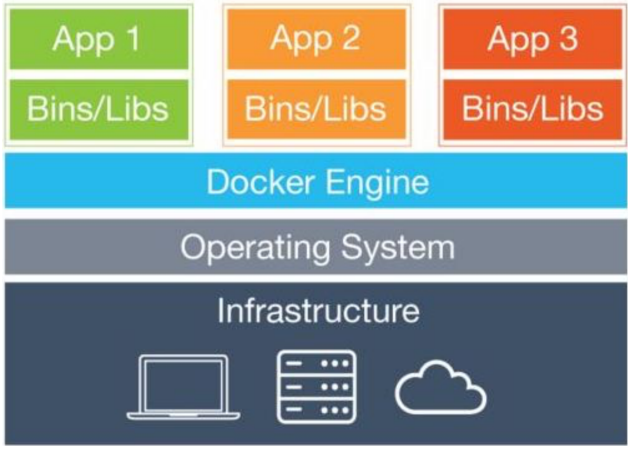
opposite problem. We are using a ton of resources to do the exact same stuff multiple times. What if we could redirect these resources so that the applications themselves could have more to work with?

6.4 Containers

A few years after the concept of virtual machines was developed, some very smart people out there came up with the concept of "containerization." Containerization is another type of virtualization, but instead of making multiple guest operating systems on top of an existing operating systems like virtual machines do, containerization divides the resources of the host operating system. Containerization involves creating "containers," which are essentially sandboxes for applications to run in. By sandbox, I mean that it utilizes resources of the host operating system, but cannot impact anything outside of it. This provides a safe space for applications to run in. Each container gets its own chunk of everything that it could need from the host operating system—file system, system tools, network stack, etc.

6.5 Docker

Docker is an open source project that began in 2010 with the goal of providing abstraction to containers. Using Docker, application developers are able to quickly and easily containerize their applications. In order to use Docker, the host machine runs the Docker Engine, which handles the management and creation of containers. Here is a picture showing how containerization with Docker works (compare this to the Virtual Machine Picture):



As you can see from the image, we no longer have to make multiple copies of the same operating system to run multiple applications. Instead, everything shares the resources of the host operating system. Docker is designed to be run on Linux machines (since containers are dependent on the operating system they are running on), but they also provide solutions that will run on Windows and Linux using very very lightweight virtual machines.

6.6 CONTAINERS ARE STATELESS

There is one key component that we can not stress enough (and will continue to remind you of). Containers are stateless. They are designed so that they can be easily destroyed and recreated (which helps make them so lightweight). This means that any files saved inside a container or any changes made to it will be erased the second that the container is terminated. If you are working on something inside a container, you have to make sure you save it outside of the container before shutting the container down.

7 Addendum: Why should I care about docker?

What a great question. Previously, we had our students install Virtual Machines in order to complete their assignments for this class. For some of them, it worked fine and was really easy. Most had at least one of the following issues:

- VMs crashing all the time
- VMs not connecting to internet
- VMs running incredibly slow
- Issues installing tools that the class needs
- Random issues that only apply to a single student and can only be solved by deleting the VM and creating a new one (which takes forever)
- A ton more

As a result, we are using Docker to mimic a VM. This isn't exactly how Docker is meant to be used, but it will provide a lot of benefits including:

- Serious performance improvements
- No networking issues because you won't use the network with it
- Seamless integration with your host operating machine
- Comes pre-installed with all the tools that you will need
- Since containers are stateless, if you have any problems, all you have to do is delete it and create a new one, which takes about 5 seconds