Docker Quick Start

Competencies

By the end of the Docker QS, you should be able to answer the following:

- Why do we use Docker in the bootcamp?
- Loosely, what is the difference between an image and a container?
- How do you pull a new image?
- How do you visualize porting? When starting a container with `docker run`, what is the syntax for porting?
- How do you visualize bind mounting? When starting a container with `docker run`, what is the syntax for bind mounting?
- Loosely, when starting a container with `docker run`, what does using the flag `-it` do?
- Loosely, when starting a container with `docker run`, what does using the flag `--rm` do?
- Suppose you start a docker container; describe roughly how to stop and restart the same container? Is this affected by using `--rm` when starting the container? Are changes to the container recorded on restarting?
- How can you find which containers are still active? How to you deactivate or remove them?
- Does MySQL require Docker? Does Linux require Docker? Does TensorFlow require Docker?

Installation

Mac:

Installation download and documentation here: https://docs.docker.com/docker-for-mac/install/

Windows:

Note: *Docker Desktop usually only runs natively on Windows 10 Pro, Enterprise or Education. Other versions, such as Windows A (A<10) and Windows 10 Home edition need to use Docker Toolbox.* The main difference seems to deal with virtualization; whether or not you are capable of enabling Hyper-V for virtualization (available on 10 Pro/Enterprise/Education), or need to use an installation VirtualBox with virtualization enabled (10 Home, or older editions). It's important to read the documentation in each case to make sure your installation of Docker functions as expected.

Installing Docker Desktop on Windows (10 Pro, Enterprise, or Education)

Installation download and documentation here: https://docs.docker.com/docker-for-windows/install/

Installing Docker Toolbox on Windows (10 Home, or earlier editions)

The documentation about installing Docker Toolbox for Windows can be found here: https://docs.docker.com/toolbox/toolbox_install_windows/. You will notice that you must have virtualization enabled, which may require adjustment to the BIOS. WARNING! Changing your BIOS can fundamentally affect the way your computer functions, do so very carefully, only making changes that you need to for Docker. For help with enabling virutalization, here is a reference: https://support.bluestacks.com/hc/en-us/articles/115003174386-How-can-I-enable-virtualization-VT-on-my-PC-

Linux:

Below are links to the installation packages and instructions for your specific flavor of Linux:

- Ubuntu: https://docs.docker.com/install/linux/docker-ce/ubuntu/
- Debian: https://docs.docker.com/install/linux/docker-ce/debian/
- Fedora: https://docs.docker.com/install/linux/docker-ce/fedora/

- CentOS: https://docs.docker.com/install/linux/docker-ce/centos/
- Binaries: https://docs.docker.com/install/linux/docker-ce/binaries/

Remarks on Bind Mounts / Volumes

Using bind mounts / volumes allow us to share a directory between our local machine and a container. Roughly, when starting a new container with `docker run`, we can add a `-v` flag to indicate we would like to "bind" a local directory and a directory in the container, *but* when we choose which local directory to use, we need to be mindful of syntax and the permissions Docker has to access that local directory. In pseudo-code, the correct notation is

```
docker run -v <path-to-local-directory>:<path-to-container-directory>
```

This command will bind your local directory to the container directory, and even will create the container directory at the path you specify if it doesn't already exist, as long as the mounting is done correctly.

While the path to the container directory is consistent across machines, the notation and permissions of the path to the local directory differs and takes some consideration, particularly for Windows users.

Mac & Linux:

Within Docker Desktop, you can specify which directories are allowed to be accessed by Docker containers. When you do specify a directory, then all sub-directories will also be permitted, so typically having something like `/Users` in Mac is general enough for our purposes. To look at and add/remove permitted directories (at least in Mac), you navigate to Docker Desktop Preferences, and from there to Resources > File Sharing.

The notation for mounting when initializing a run of a container is fairly straight-forward. For example, if the local path to the desired shared directory is `/Users/JaneDoe/local_folder` and we want to bind that to the container folder `/tmp/Janes_big_adventure`, then the command will be something like

```
docker run -it -v "/Users/JaneDoe/local_folder":"/tmp/Janes_big_adventure" nycdsa/linux-toolkits
```

Of course, you may be using other flags also not represented in the above example.

Windows:

By default, it is apparent that by hard-coded default, Docker has permission to mount folders within `C:\Users`, but not elsewhere. It is possible to add other permitted directories, but we will work under the assumption that your desired mounted folder lands within `C:\Users`. Because of the distinction of the foward slash `/` used in almost all non-Windows systems and urls when navigating through directories, rather than the backslash `\` used in Windows, we have a little bit of funny notation for specifying the local path. The main difference is that instead of `C:\Users`, we will use `//c/Users` as the base path and make sure to use forward slashes throughout. So, if the local path we want to mount is `C:\Users\JaneDoe\local_folder` then the notation to mount this to the container directory `/tmp/Janes_big_adventure` is

```
docker run -it -v "//c/Users/JaneDoe/local_folder":"/tmp/Janes_big_adventure" nycdsa/linux-toolkits
```

Of course, you may be using other flags also not represented in the above example.