Docker notes 01: First steps with docker

Docker: Hello world

A two-line check for whether docker works to the degree needed on your machine is to pull the official `hello-world` container. You have just pulled an image (`hello-world`) from the public docker repositories.

```
sudo docker pull hello-world
sudo docker run hello-world
```

Expected output:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Not there yet? Installation

This project was built on an ubuntu virtual machine. Docker was installed using this set of commands:

```
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg lsb-release
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/key
sudo ls /etc/apt/keyrings/
echo   "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
sudo docker run hello-world
```

If you are on a different distribution, follow the official installation steps for docker engine

If you are on a windows machine, you may need to install docker desktop

Explanation

The docker repository contains the pre-made images most applications use as their baseline. For example, `ubuntu:20.04` and `alpine:latest` provide minimal linux environments to deploy software. A `Dockerfile` is used to maintain the deploy sequence.

Say Hello Back

Instead of pulling a completely pre-built contianer, build your own using a linux baseline and compiled java code:

Java applicaition:

```java
// Main.java
public class Main {
    public static void main(String[] args){
        System.out.println("Hello World!");
    }
}
```

Dockerfile:

This sequence of docker commands pulls a linux image, installs the java compiler (`default-jdk`), compiles the main class using `javac`, and finally runs `java main`.

```
FROM ubuntu:20.04

RUN apt update
RUN apt install default-jdk -y
COPY . .
RUN javac Main.java

CMD ["java", "Main"]
```

Shell:

This command builds the container according to the dockerfile in the current directory (`.`) and gives it a tag (name) of hello. A tag is represented with `-t`.

```
docker build -t hello .
docker run hello
#> Hello World!
```