**Dive into Docker**

* Easy to install and run software without worrying about underlying dependencies
* Docker is platform or eco system that supports running containers

1. Docker Client (CLI) : Command to interact with docker server
2. Server (Daemon) : responsible for creating images and running containers
3. Machines
4. Images : Single file contains all the dependencies and configuration to run the program.

FS snapshot + startup command

1. Hub
2. compose

Container:

* Instance of images,
* separate hardware memory.
* A process or list of processes that have a grouping of resources specifically for it.

How does it work?

* Check the copy in the local image directory. If not, download it from the hub.
* Kernel is responsible for allotting the hardware resources
* Namespacing : Isolating the resources per process
* Control groups: Limit amount of resources user per process
* NS and Control group specific for linux kernel not for windows or mac.
* Docker installation behind the scene runs a linux virtual machine which allottes the NS and CG.

**Manipulating with Docker client**

* *Docker run <image-name> <default-command>*
* Docker run = docker create + docker start
* Docker ps –all
* Docker ps
* We can restart the container using hash code.
* *Docker system prune*
* *Docker logs <id>*
* *Docker stop <id> :SIGTERM, gracefully. After 10sec SIGKILL*
* *Dock er kill <id>: SIGKILL*
* *Docker exec -it <id> <command>*
* *Docker run -it <image> sh*

*Docker image runs in an isolated env.*

**Building Custom images**

DockerFile: Configuration to define how the container should behave.

Creating docker file

1. Specify the base image
2. Run commands to install additional programs
3. Specify the startup command

*Docker build .*

*Rebuild with cache*: Each step snap-shot is saved in phase by phase manner. So that docker can reuse the snapshot again for the future process. The reordering of commands will lead to a complete new run.

Docker build -t <your-docker-id/project-name:version> .

Manual generating : docker commit -c ‘<run-command>’ <id>

**Project Outline**

Sample docker file of nodeJS app

*FROM node:alpine*

*WORKDIR /usr/app*

*COPY ./ ./*

*RUN npm install*

*CMD [“npm”, “start”]*

* Container port mapping: Docker run -p 8080:8080 <id>
* WORKDIR : any command execute relative to this directory
* Minimizing the rebuilds: make sure reuse existing layer images wisely (COPY ./package.json .)