# Development

Meeting Docker

# Outline

- Editors
- Development Environment
- Local, VM, Docker
- First Steps with Docker

# Editors

- Need to write files
  - 
- No requirement to use product X or Y
  - You should like it
  - Have features you like
    - Code highlighting
    - Useful templates
- I frequently change editors
  - Free
  - Feature
  - Visual Studio Code

# Software Development

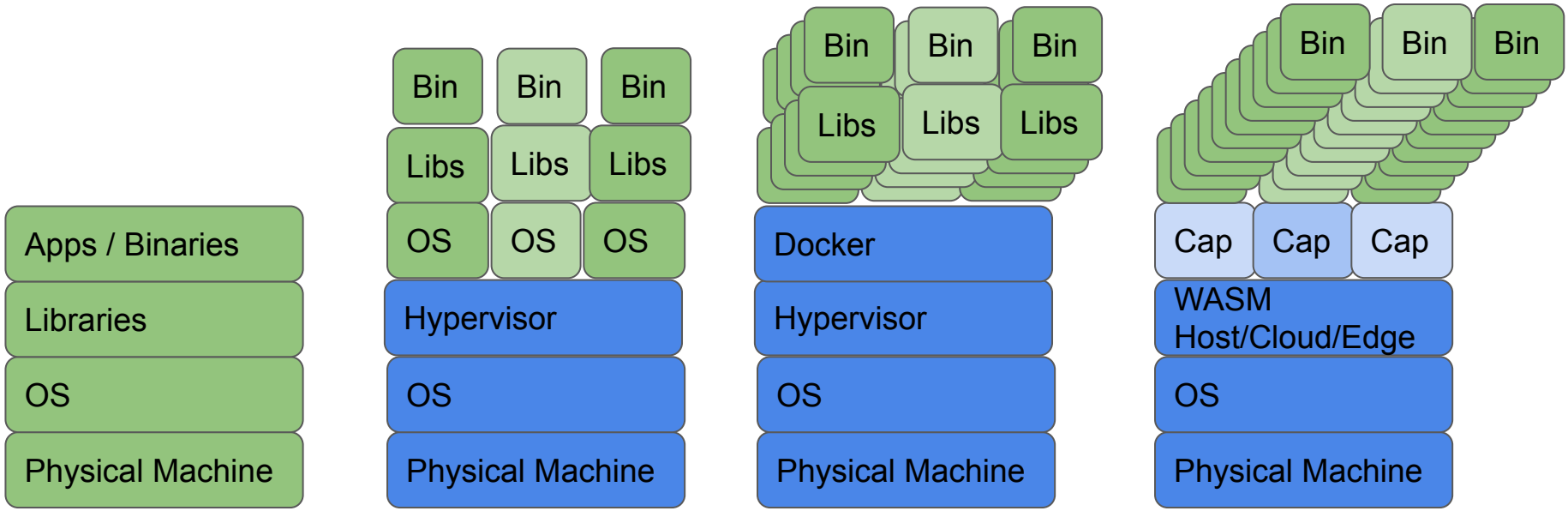
- Development Environment, Production Environment
  - DEV == PROD?

- Need for different languages/frameworks
- Need to run networks

# Developing

- Local
- VM
- Containers -> **Docker**
- WebAssembly

# Development




**Computer:**
**PC, Server, …**

**Virtual machine**

**Container**

**WebAssembly**

© Ralph Deters - CMPT 353 -2025

# Local

- Create local environment
  - Install servers/libraries
  - 
- Advantages:
  - You are in control
  - Customize everything
  - Low  overhead
  - 
- BUT

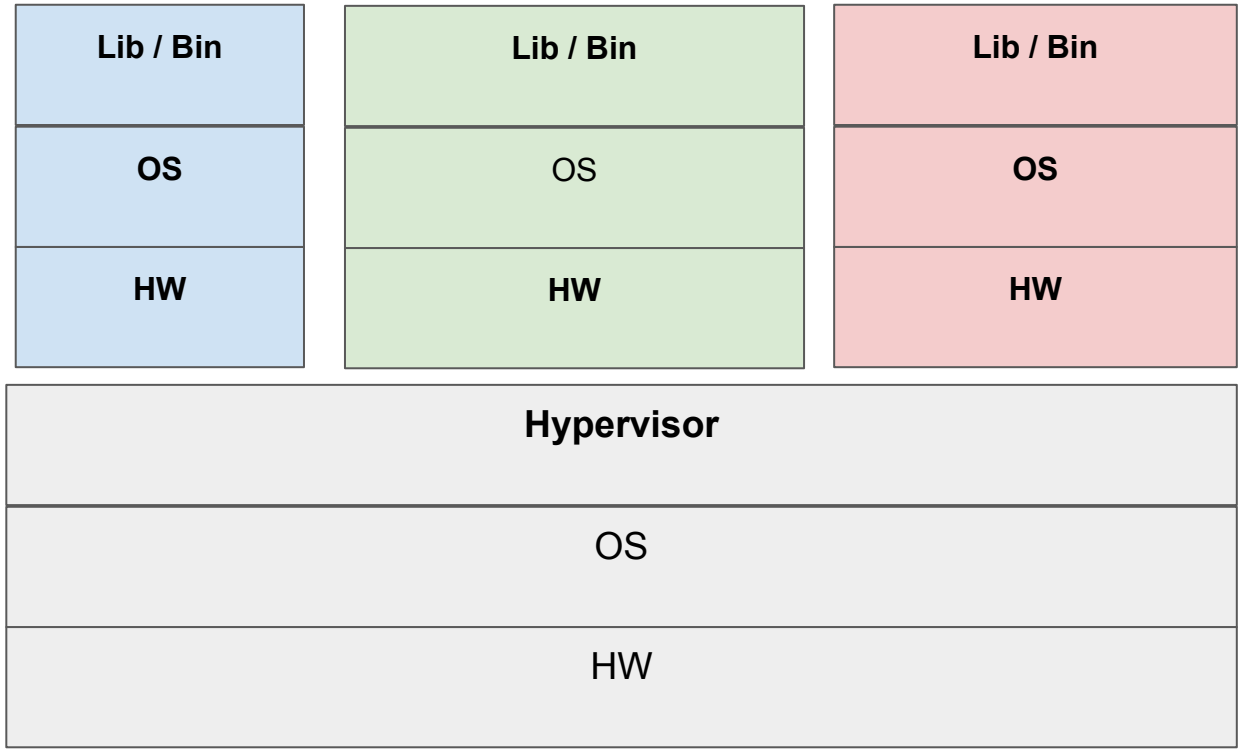
# Disadvantages

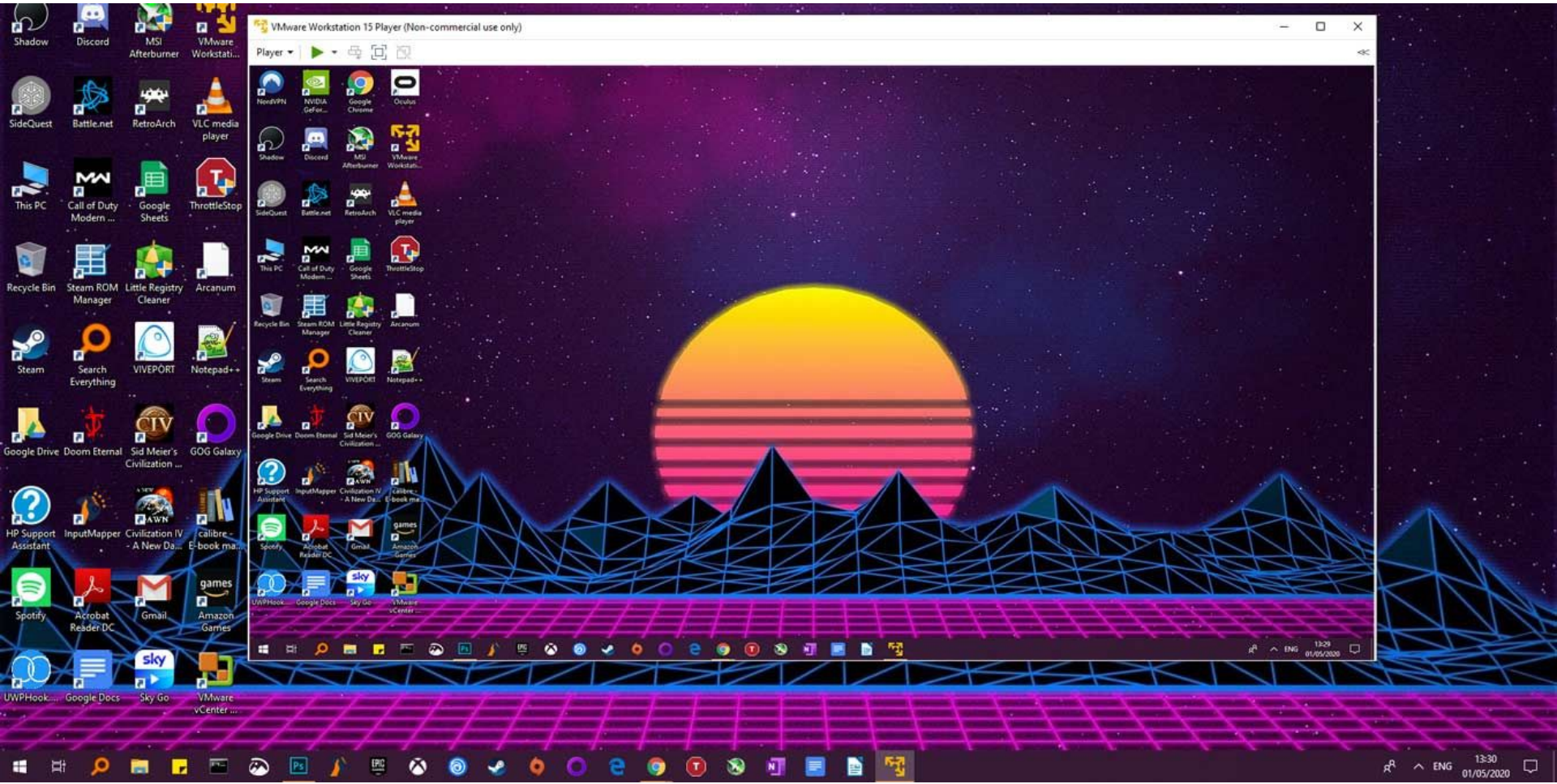
- Installations
- Library versions
  - Different projects may need different versions/subversions
- Directory separator
  - / A / B
  - \ A \ B
- Case Sensitivity
  - Does upper/lower case matter?
- File encoding
- Permissions …

Yes your IDE can help and yes code repository can help …..

# Virtual Machines

- Virtualize computers
  - Simulate machines




VMware Workstation 15 Player (Non-commercial use only)
Player
Shadow
Discord
MSI Afterburner
VMware Workstati...
SideQuest
Battle.net
RetroArch
VLC media player
This PC
Call of Duty Modern ...
Google Sheets
ThrottleStop
Recycle Bin
Steam ROM Manager
Little Registry Cleaner
Arcanum
Steam
Search Everything
VIVEPORT
Notepad++
Google Drive
Doom Eternal
Sid Meier's Civilization ...
GOG Galaxy
HP Support Assistant
InputMapper
Civilization IV - A New Da...
calibre - E-book ma...
Spotify
Acrobat Reader DC
Gmail
Amazon Games
UWPHook...
Google Docs
Sky Go
VMware vCenter ...
ENG
13:30
01/05/2020


Preview File Edit View Go Tools Bookmarks Window Help
Mon 2:09 AM
Terminal — telnet — 80×24
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Hercules version 3.06 built on Jan 30 2009 21:35:31
running on gojira.local (Darwin-9.6.0.Darwin Kernel Version 9.6.0: Mon Nov 24 17:37:08 i386 MP=2)
Connected to device 0:001F
VM/370 VERSION 06 LEVEL 00 PLC 0029; 09/07/85 09:42:40
NOW 08:06:54 PST MONDAY 02/09/09
CHANGE TOD CLOCK (YES NO) :HHCTE006A Enter input for console device 001F
NO
08:06:58 START ((COLD WARM CKPT FORCE) (DRAIN)) (SHUTDOWN) :HHCTE006A Enter input for console device 001F
COLD
08:07:00 AUTO LOGON *** OPERATOR USERS = 001 BY SYSTEM
DMKCP1957I STOR 02048K, NUC 180K, DYN 01712K, TRA 032K, FREE 0124K, V=R 00000K
08:07:00 FILES: NO RDR, NO PRT, NO PUN
enable all
08:07:04 08:07:04 COMMAND COMPLETE
Terminal — hercules — 80×57
Hercules CPU0000: 0% S/370
Peripherals
U Addr Modl Type Assignment
A 000C 3505 RDR * intrq
B 000D 3525 PCH punch00d.txt ascii
C 000E 1403 PRT print00e.txt crlf
D 001F 3215 CON 127.0.0.1
E 0040 3270 DSP 127.0.0.1
F 0041 3270 DSP
G 0580 3420 TAPE dmkddr.aws [1:00000000]
H 0581 3420 TAPE VMREL6.ddr.aws [1:00000
I 0582 3420 TAPE CPR6L0.ddr.aws [1:00000
J 0131 3330 DASD VMREL6.3330-1 [404 cyls
K 0132 3330 DASD CPR6L0.3330-1 [404 cyls
PSW
GPR CR AR FPR
ADDRESS: DATA:
STO DIS RST
MIPS SIO/s
STR STP EXT IPL PWR
CPU
0
x3270-2 127.0.0.1:3270
File Options
VM/370 ONLINE
RUNNING

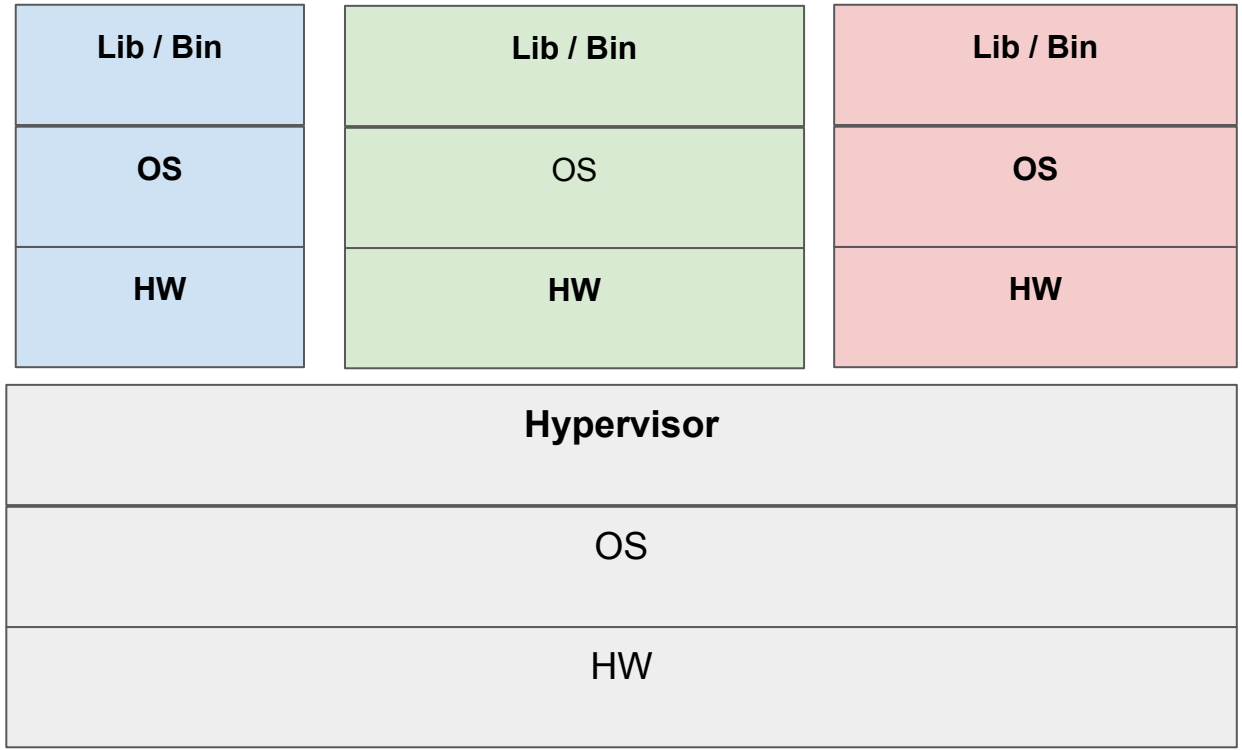
# Virtual Machines

- Create virtual machines that match production environment !
- Advantages:
  - Total isolation
  - Replication of production environment
  - Widely used
  -
- BUT
  - Resource intensive => expensive
    - Why???

# Containers !!

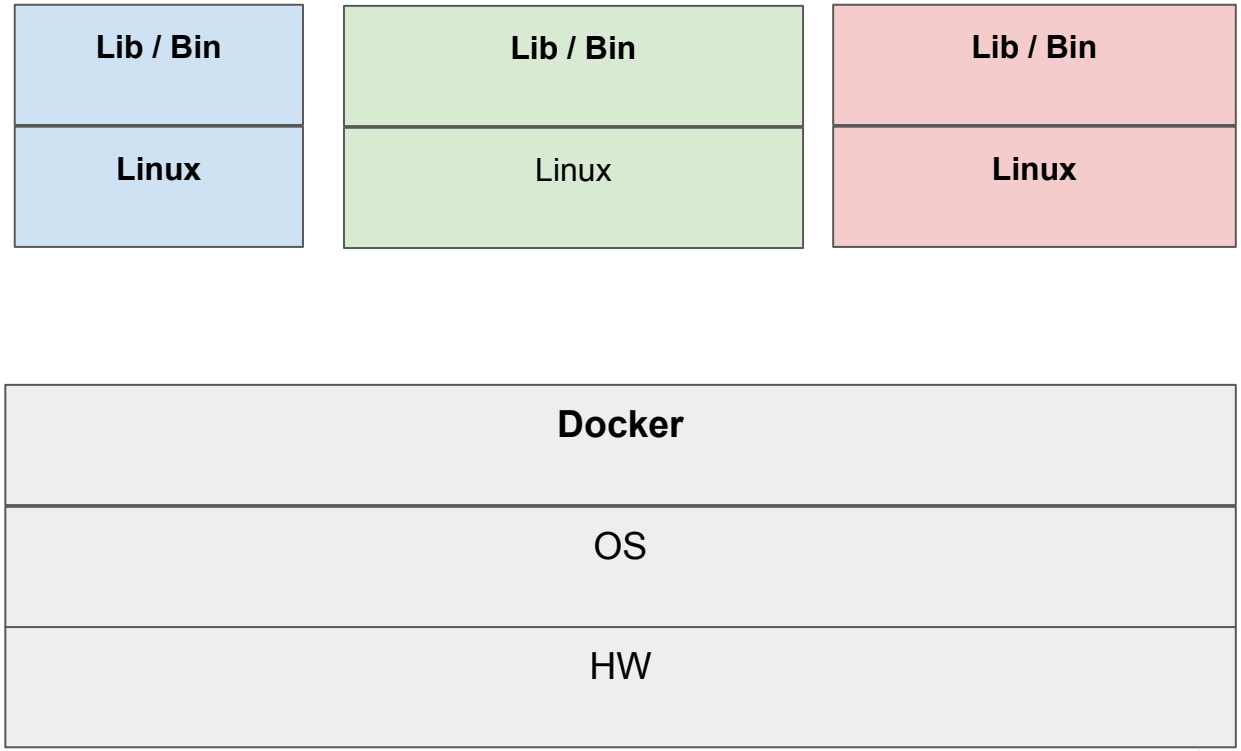
- Define compute environments => containers
- “containerize applications”
- Many Container Solutions
- Docker is most popular
  - Cloud Vendors have own solutions
    - Docker compatible
- 2 basic options
  - Docker Playground:
    - https://labs.play-with-docker.com/
    - Need to create a free account
  - Local Install
    - https://www.docker.com/

# VM




© Ralph Deters - CMPT 353 -2025

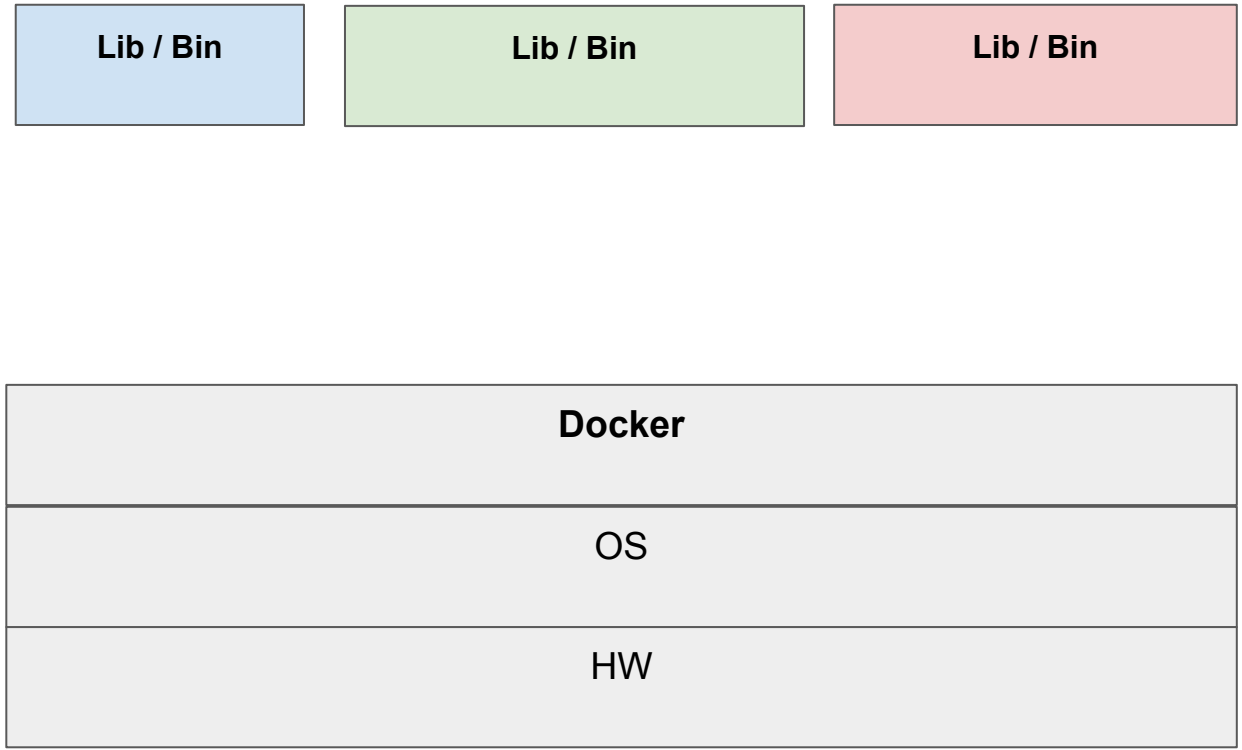
# Container






© Ralph Deters - CMPT 353 -2025

# Container




© Ralph Deters - CMPT 353 -2025

# Docker

- Widely used approach
- Active community
- Key concepts
  - Dockerfile
  - Image
  - Container
- How to get docker?
  - https://www.docker.com/products/docker-desktop
- What kind of “images” are there?
  - https://hub.docker.com/search?q=&type=image

# Steps

1) Get base-image with **pull** command
    a) Something that is close to what you want
2) Write a Dockerfile (script) to modify/enrich base image
3) Create new image with **build** command
4) Run new image with **run** command

Let’s install a Node,js server with docker

# Let’s start

- Get image
  - `docker pull node:latest`
- Write your Dockerfile

# Dockerfile

```
FROM node:latest

EXPOSE 8080

CMD ["/bin/bash"]
```

# Why Expose???

- Docker containers are locked down
- You have to actively enable capabilities (e.g. allow use of port 8080)

...

- Build our own image
  - `docker build -t ralph/node .`

- Run Container (run instance of image)
  - `docker run -it ralph/node`

**You also have to grant capabilities to the runtime**

1) **Ports**
2) **File System**

```
docker run
-p 80:8080
-v /Users/ralph/node/nodejs:/usr/src/app/
--rm
--name a1
-it ralph/node
```

# Change run command

- Open port
  - `-p 80:8080`
- Mount directory/volume
  - `-v /Users/ralph/node/nodejs:/usr/src/app/`
- Remove container after it stopped
  - `--rm`
- Give container a name
  - `--name a1`
-

# Starting Container

1. `docker run -p 80:8080 -it ralph/node`

2. `docker run -p 80:8080  --rm -it ralph/node`

3. `docker run -p 80:8080  --rm --name a1 -it ralph/node`

4. `docker run -p 80:8080 -v /Users/ralph/code/353/node:/usr/src/app/  --rm --name a1 -it ralph/node`

# Configuring Node.js

cd **/usr/src/app/**

**Create config file of the server**

**npm init**

**Add library (package)**

**npm add express**

# Create Code for Server

First install editor

apt-get update

apt-get install nano

Now create file

# server.js

```
'use strict';

var express = require('express');  var  app = express();

app.get('/', (req, resp) => { console.log(req.originalUrl); resp.send('hello world'); });

app.get('/hello', (req, resp) => { console.log(req.originalUrl); resp.send('hello'); });

app.use('/web', express.static('pages'));

app.listen(8080);
```

# Start Server

npm start

OR

node server.js

Kill server => ctrl c

Create pages & add a file

mkdir pages

cd pages

nano ralph.txt

cd ..

node server.js

Kill server (ctrl c) & kill container (exit)

# Demo - Python

- Show a simple dockerfile
- Show a simple docker-compose.yml file
- Quick demo

# Dockerfile

```
FROM python:latest

EXPOSE 5000

WORKDIR /usr/src/app

RUN pip install Flask

CMD ["/bin/bash"]
```

...

docker build -t tag/name .

# Python file

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello_world():
        return "<p> Hello World </p>"
```

# docker-compose.yml

```
version: "3.9"
services:
 python1:
  build: .
  container_name: p1
  command: flask run --host=0.0.0.0
  ports:
   - "80:5000"
  volumes:
   - /Users/ralph/classes/436/test/python:/usr/src/app
  environment:
   - FLASK_APP=hello
   - FLASK_ENV=production
```

...

https://docs.docker.com/compose/reference/

docker-compose build

docker-compose up
docker-compose down

docker-compose up -d