Docker: An Introduction

JC Cruz

September 1, 2024

Source: github.com/yourusername/your-repo

Problem Scenerio

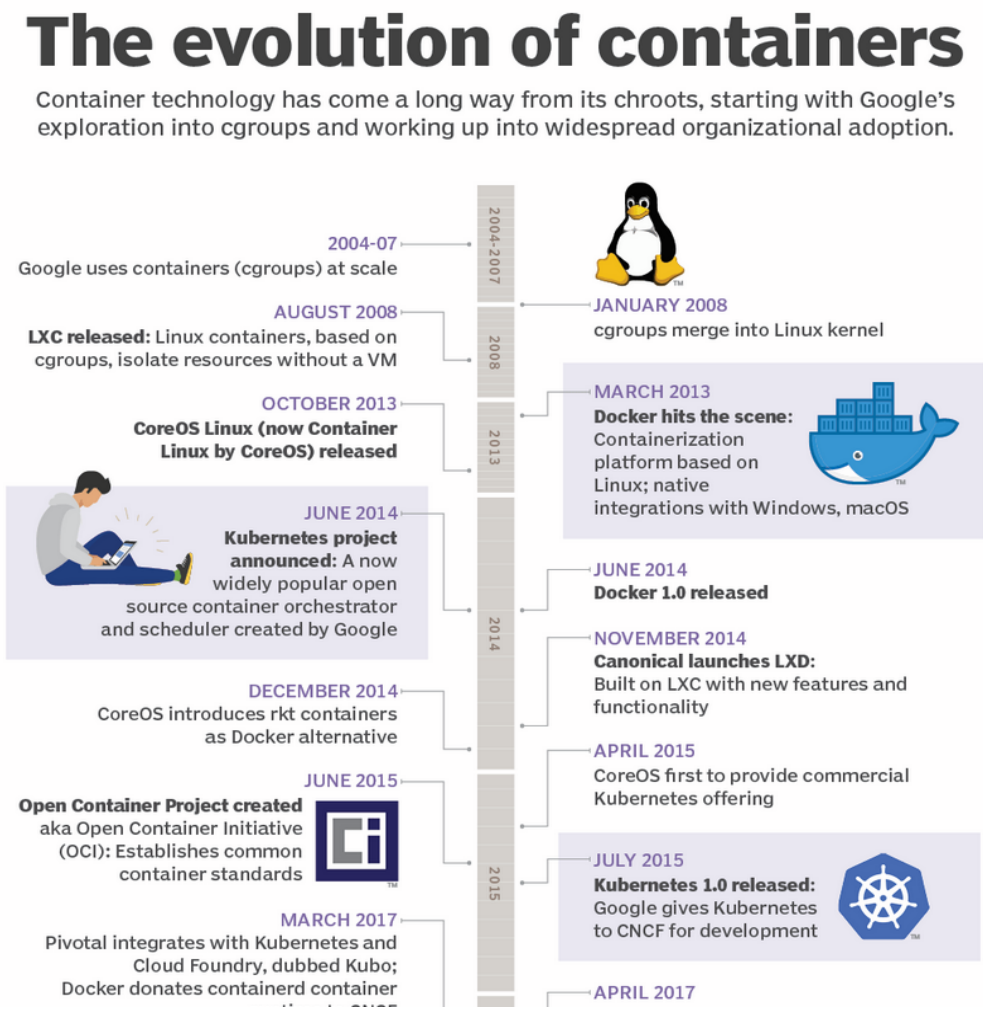
- You're on your local laptop with Windows or the latest Ubuntu image 24.04
- Your supervisor asks you to validate the results of some software from another group
- The group emails you their source code and scripts but it turns out it was developed on Ubuntu 14.04 (which released in 2014 and had EOL in April 2024) and does not work on your host PC
- What to do now?

Solutions

1. Setup and configure on some old HDD or SSD
   - Takes time, hardware
   - No guarantee of out-of-box compatibility
2. Boot into VMware with the correct OS
   - No guarantee of out-of-box compatibility
   - Can be pretty slow
3. Use Docker
   - Quick setup
   - Consistent environment across different machines
   - Lightweight compared to VMs

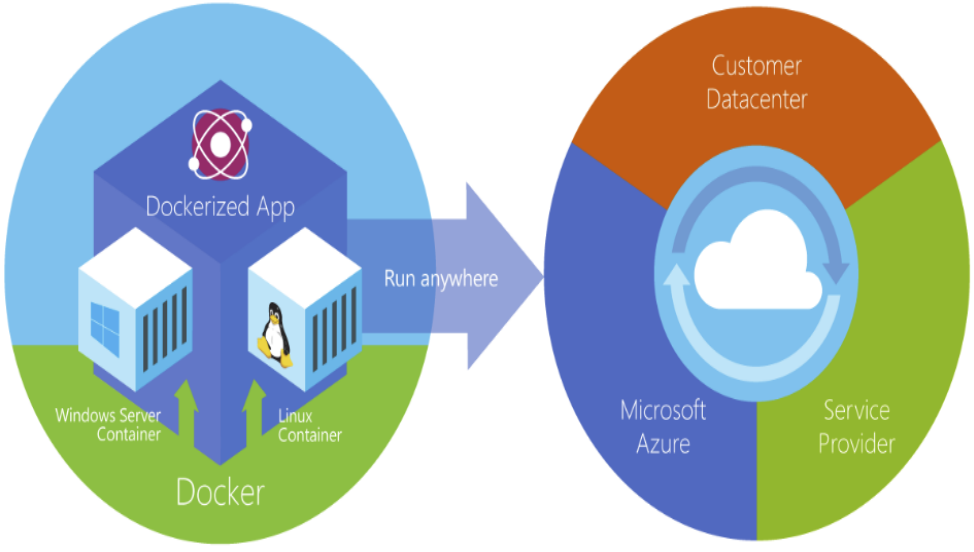
Docker Background

- Launched in 2013
- Open-source platform for containerization
- Allows applications to run in isolated environments called containers
- Uses host OS kernel, making it more efficient than VMs



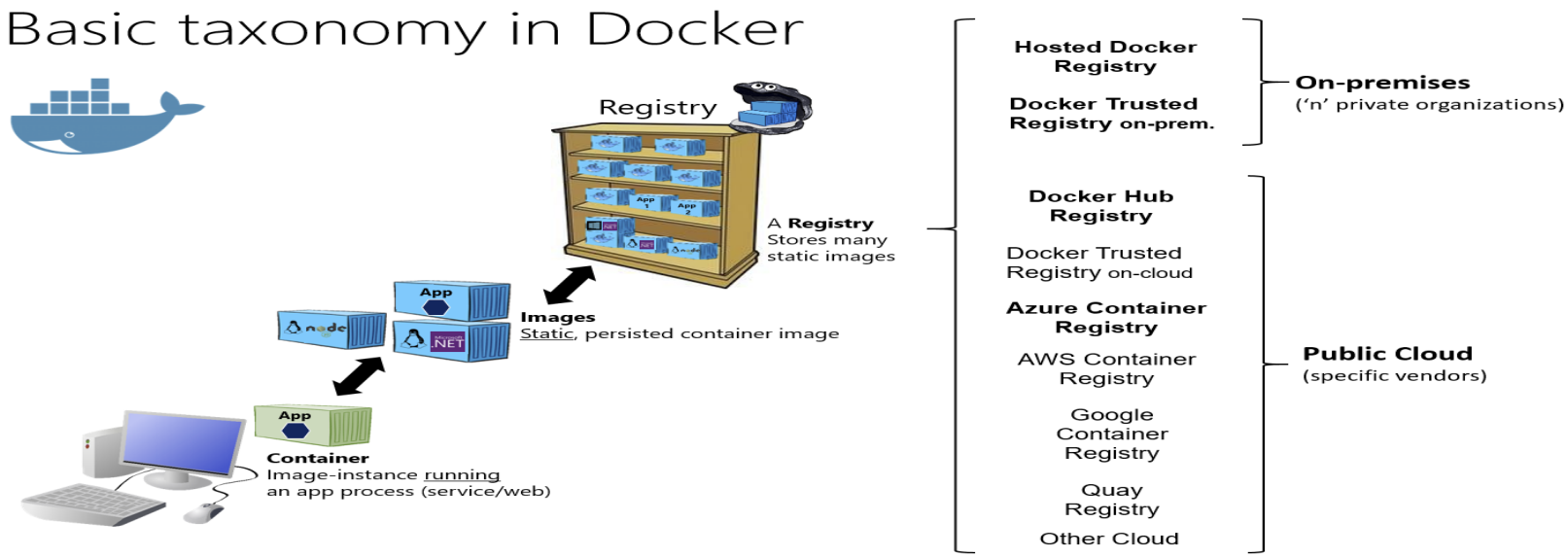
Benefits of Docker

- Avoid the problem of "it runs on my machine"
- Faster deployment and scaling
- Better resource utilization compared to VMs
- Simplified dependency management
- Easier application isolation and security



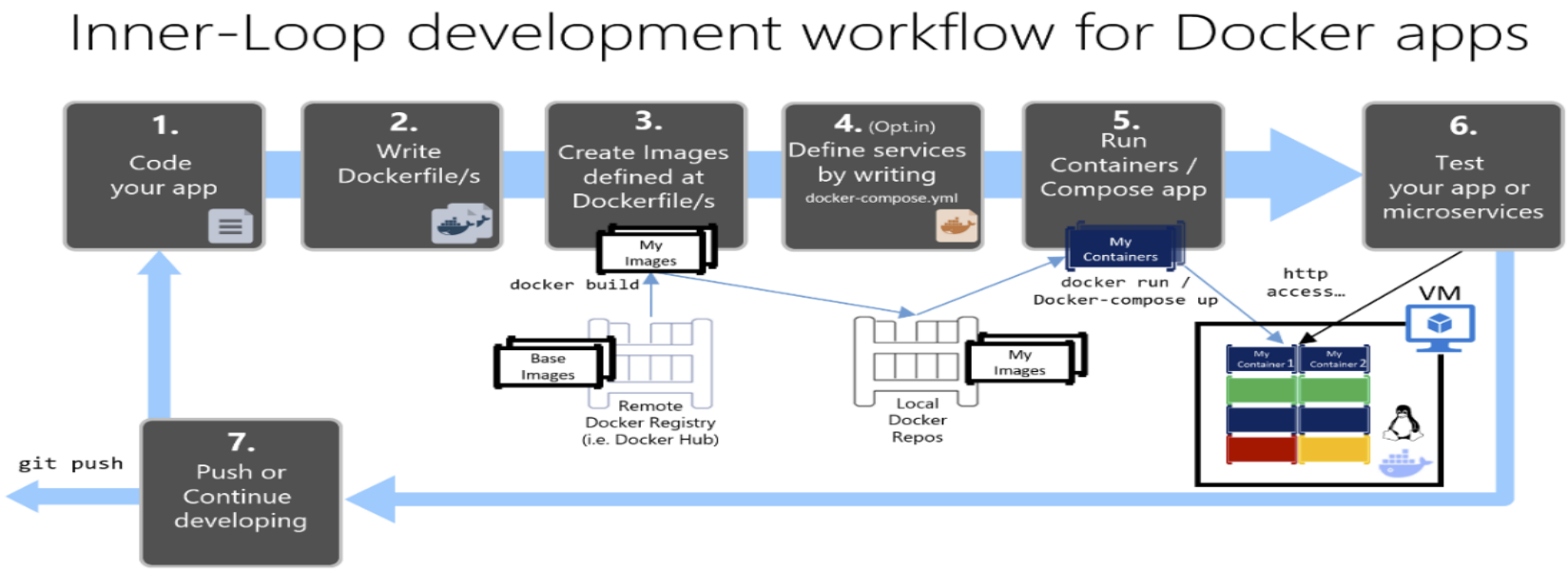
Docker Architecture

- Docker Engine: Core component that builds and runs containers
- Docker Client: CLI tool for interacting with Docker
- Docker Daemon: Background service managing Docker objects
- Docker Registry: Storage for Docker images (e.g., Docker Hub)
- Docker Objects: Images, containers, networks, volumes



Important File Types

- Dockerfile (Dockerfile)
  - Defines a single container
  - Used to build custom images
- Docker Compose (docker-compose.yaml)
  - Orchestrates multiple containers
  - Manages application-wide settings and networking



Docker Commands

- `docker build`: Build an image from a Dockerfile
- `docker run`: Create and start a container from an image
- `docker pull`: Download an image from a registry
- `docker push`: Upload an image to a registry
- `docker ps`: List running containers
- `docker images`: List available images
- `docker stop`: Stop a running container
- `docker rm`: Remove a container
- `docker rmi`: Remove an image

Docker Compose Commands

- `docker-compose up`: Create and start containers
- `docker-compose down`: Stop and remove containers, networks, images, and volumes
- `docker-compose build`: Build or rebuild services
- `docker-compose ps`: List containers
- `docker-compose logs`: View output from containers
- `docker-compose exec`: Run commands in a running container

Example: Python Development

- Github repository: https://github.com/jc-cr/python_dev_docker
- Directory structure:

```
.
├── common_stat_distributions_figs
│   ├── Dockerfile
│   ├── main.py
│   └── requirements.txt
├── docker-compose.yaml
├── program_outputs
└── README.md
```

Dockerfile

```
# Use an official Python runtime as a parent image
FROM python:3.8.19-bookworm

# Avoid prompts from apt
ENV DEBIAN_FRONTEND=noninteractive

# Set the working directory in the container
WORKDIR /container_workspace

# Install system dependencies
RUN apt-get update && apt-get install -y \
    gcc \
    && rm -rf /var/lib/apt/lists/*

# Copy the current directory contents into the container
COPY . /container_workspace/

# Install any needed packages specified in requirements.txt
COPY requirements.txt /container_workspace/
RUN pip install --no-cache-dir -r requirements.txt

# Run main.py when the container launches
CMD ["python", "main.py"]
```

docker-compose.yaml

```
version: '3'
services:
  python_plotting_script:
    build: ./common_stat_distributions_figs
    volumes:
      - ./program_outputs:/container_workspace/program_outputs
```

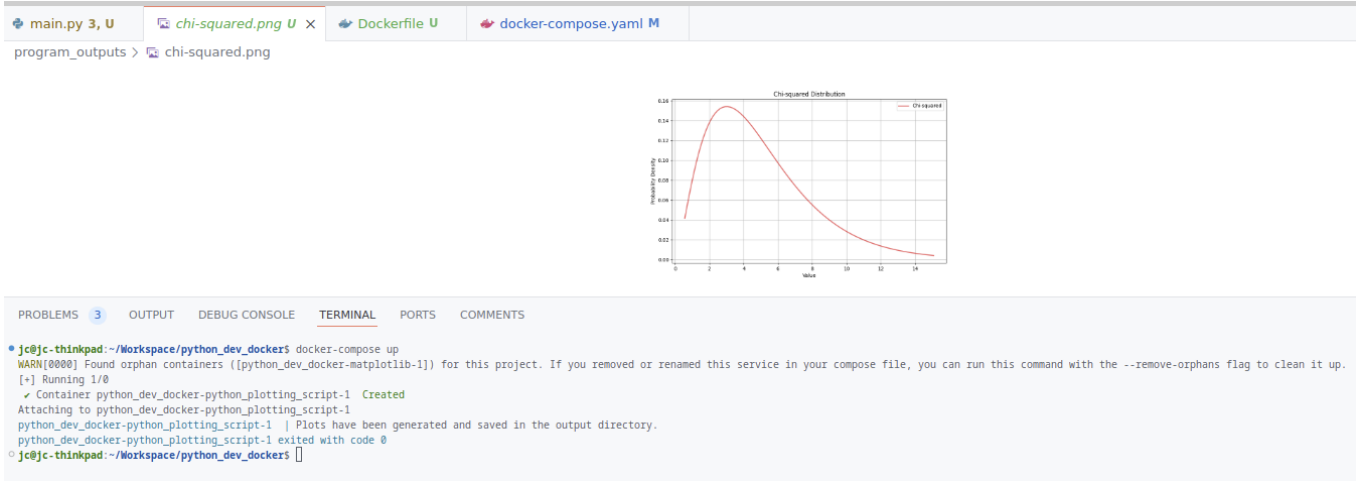
Building

- docker-compose build

```
jc@jc-thinkpad:~/Workspace/python_dev_docker$ docker-compose build
[+] Building 98.3s (11/11) FINISHED                                                                                    docker:default
 => [python_plotting_script internal] load .dockerignore                                                                        0.0s
 => => transferring context: 2B                                                                                                 0.0s
 => [python_plotting_script internal] load build definition from Dockerfile                                                     0.0s
 => => transferring dockerfile: 709B                                                                                            0.0s
 => [python_plotting_script internal] load metadata for docker.io/library/python:3.8.19-bookworm                               0.9s
 => [python_plotting_script internal] load build context                                                                        0.0s
 => => transferring context: 1.79kB                                                                                             0.0s
 => CACHED [python_plotting_script 1/6] FROM docker.io/library/python:3.8.19-bookworm@sha256:0747706cabf2997d3b0f3c40ab79bfb7c564e2acc0b72c05d275674ba12fd015   0.0s
 => [python_plotting_script 2/6] WORKDIR /container_workspace                                                                   0.0s
 => [python_plotting_script 3/6] RUN apt-get update && apt-get install -y     gcc     && rm -rf /var/lib/apt/lists/*            3.8s
 => [python_plotting_script 4/6] COPY . /container_workspace/                                                                   0.0s
 => [python_plotting_script 5/6] COPY requirements.txt /container_workspace/                                                    0.0s
 => [python_plotting_script 6/6] RUN pip install --no-cache-dir -r requirements.txt                                            92.3s
 => [python_plotting_script] exporting to image                                                                                 1.1s
 => => exporting layers                                                                                                         1.1s
 => => writing image sha256:1e5c45fb04e744e21a24b67391bec01e4abb904eaf068b29c80c861240f63f9d                                    0.0s
 => => naming to docker.io/library/python_dev_docker-python_plotting_script                                                     0.0s
jc@jc-thinkpad:~/Workspace/python_dev_docker$
```

Running

- dcoker-compose up



- What if I try to run locally?

```
jc@jc-thinkpad:~/Workspace/python_dev_docker$ python3 common_stat_distributions_figs/main.py
Traceback (most recent call last):
  File "/home/jc/Workspace/python_dev_docker/common_stat_distributions_figs/main.py", line 1, in <module>
    import matplotlib.pyplot as plt
ModuleNotFoundError: No module named 'matplotlib'
jc@jc-thinkpad:~/Workspace/python_dev_docker$
```