MUFFIN CONFERENCE | MusalaSoft

Achieving Continuous Delivery of Java Enterprise applications with Docker

Petyo Dimitrov

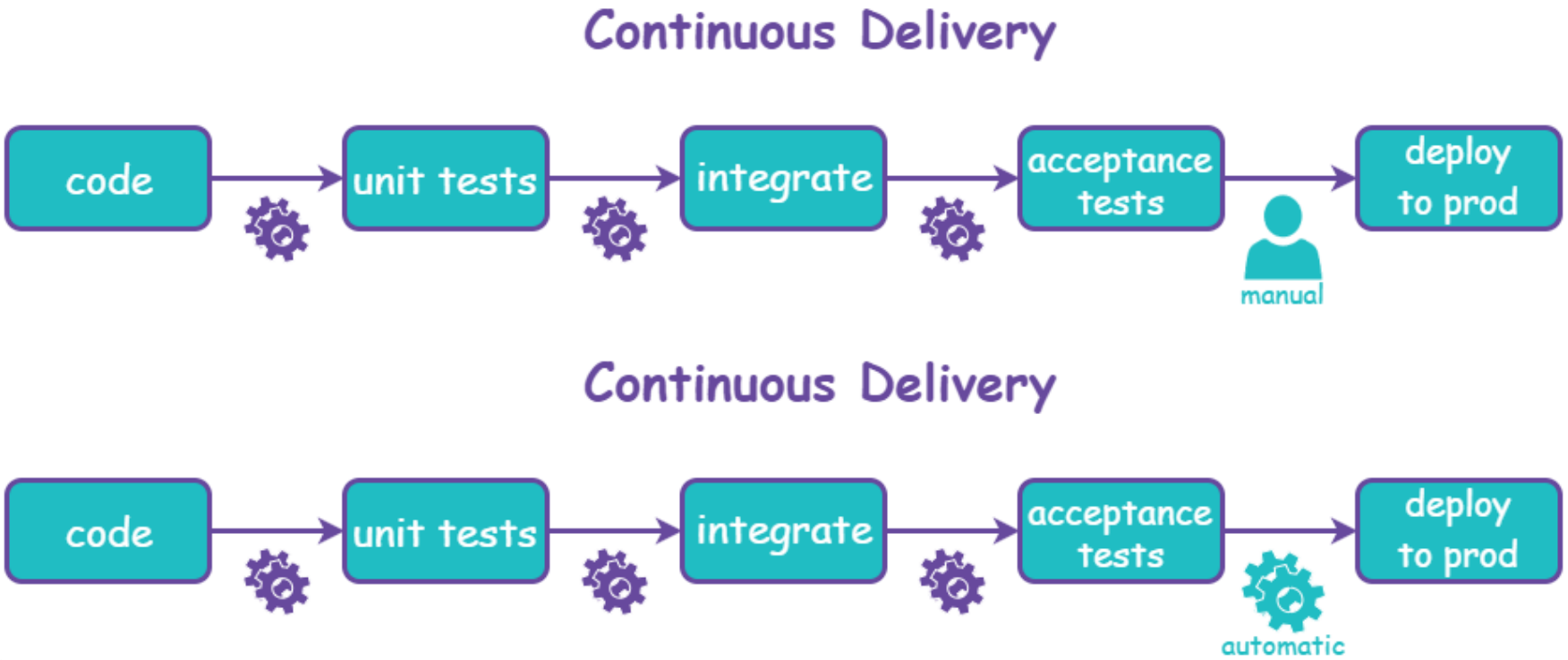
Agenda

- A typical Enterprise Java landscape
- Continuous Delivery pipeline
- Identified problems
- What are Docker containers?
- Docker "ecosystem"
- Demo
- Tips & Tricks

Dev landscape for Java EE applications

maven

Continuous Delivery pipeline



problems

inadequate packaging
of application artifacts

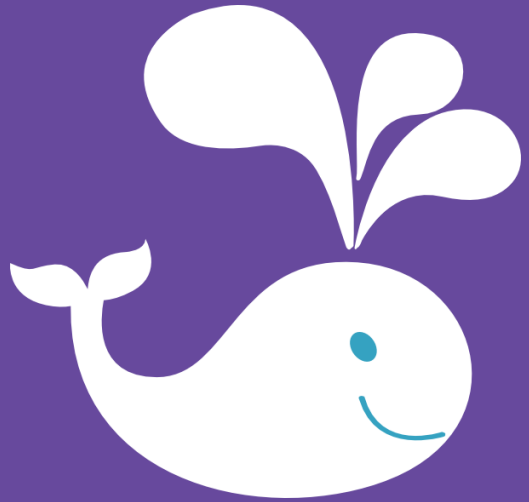
inconsistencies across *environments*

high cost

supporting multiple static environments

lack of freedom

experimenting with new languages, technologies and frameworks



docker

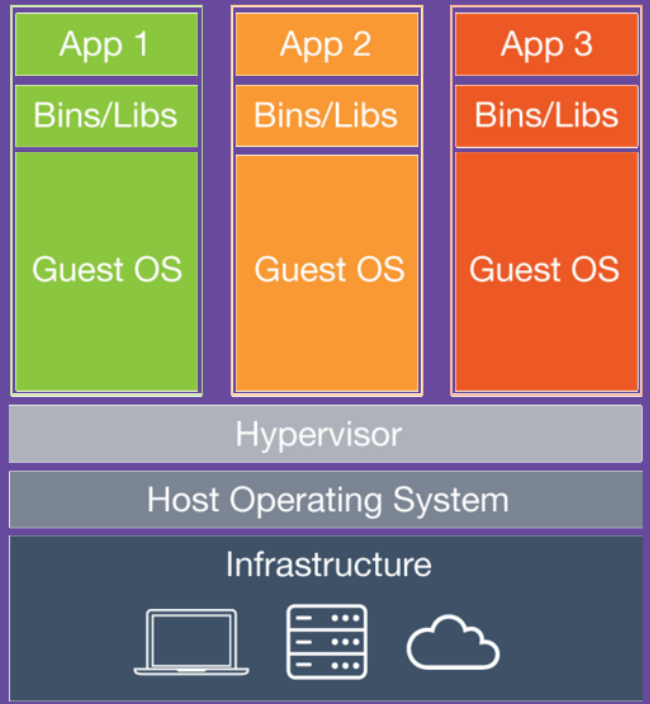
High level view of a container

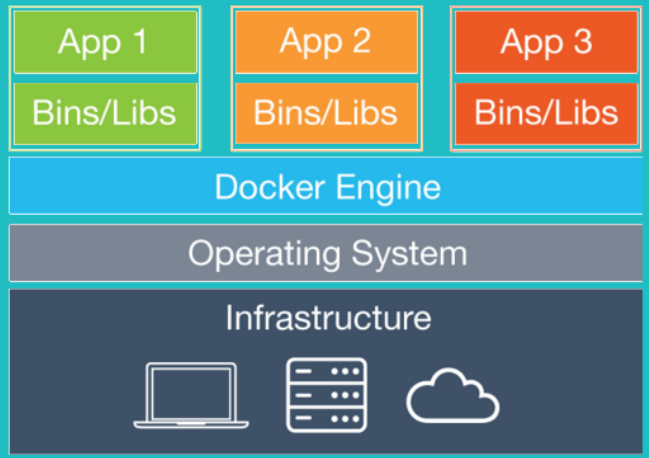
- **it is like a lightweight Virtual Machine**
- it provides:
  - own process space
  - own network interface
  - running stuff as root

Low level view of a container

- **container = "process in a box"**
- shares kernel with host → boots faster
- processes run directly on the host
- there is no device emulation
- none or little CPU, memory, network and I/O overhead

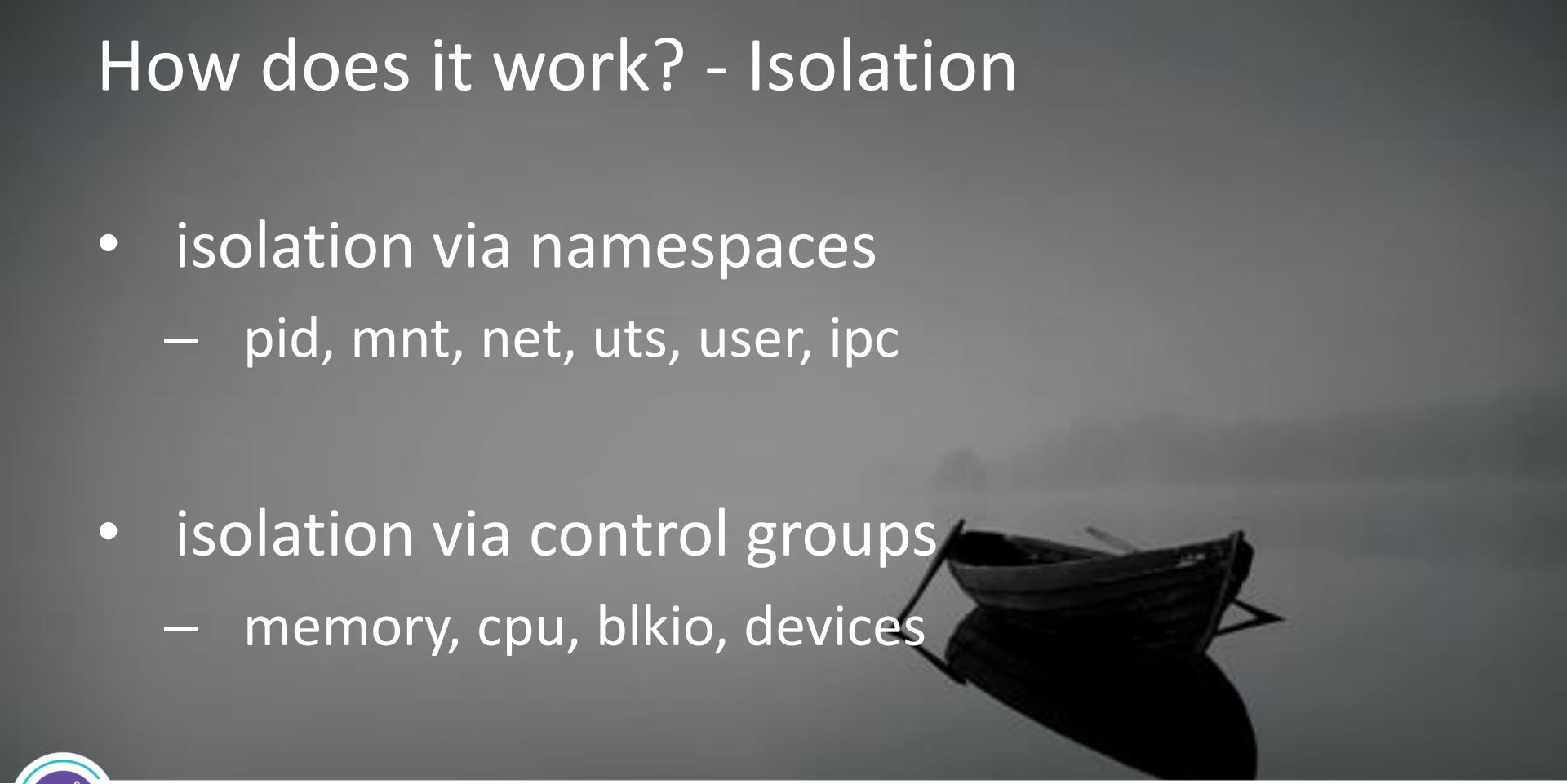
Virtual machines vs Containers



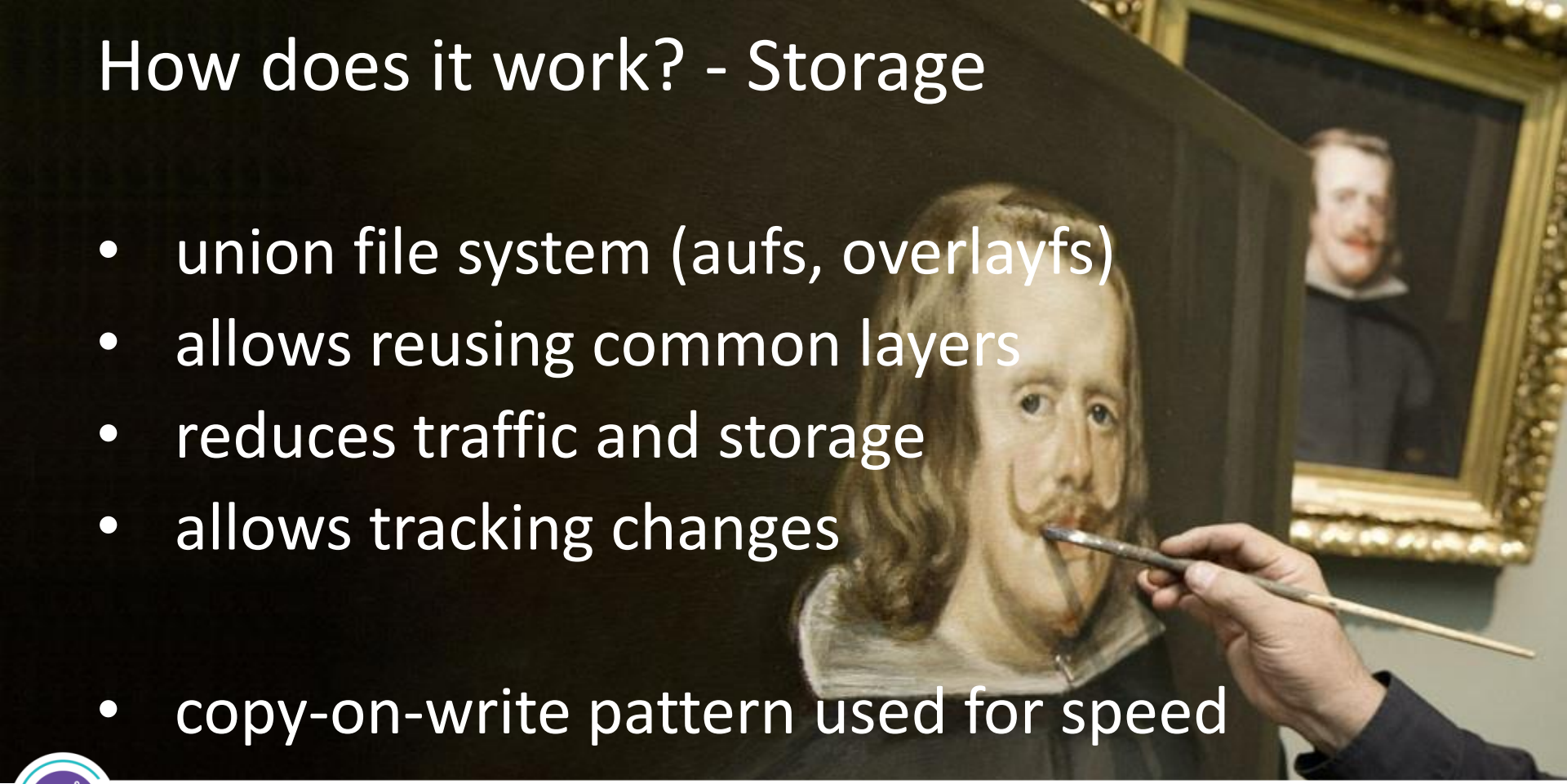


How does it work? - Isolation

- isolation via namespaces
  - pid, mnt, net, uts, user, ipc
- isolation via control groups
  - memory, cpu, blkio, devices



How does it work? - Storage



- union file system (aufs, overlayfs)
- allows reusing common layers
- reduces traffic and storage
- allows tracking changes

- copy-on-write pattern used for speed

Developers

- care about applications
- put stuff in containers
  - code & data
  - libraries
  - applications

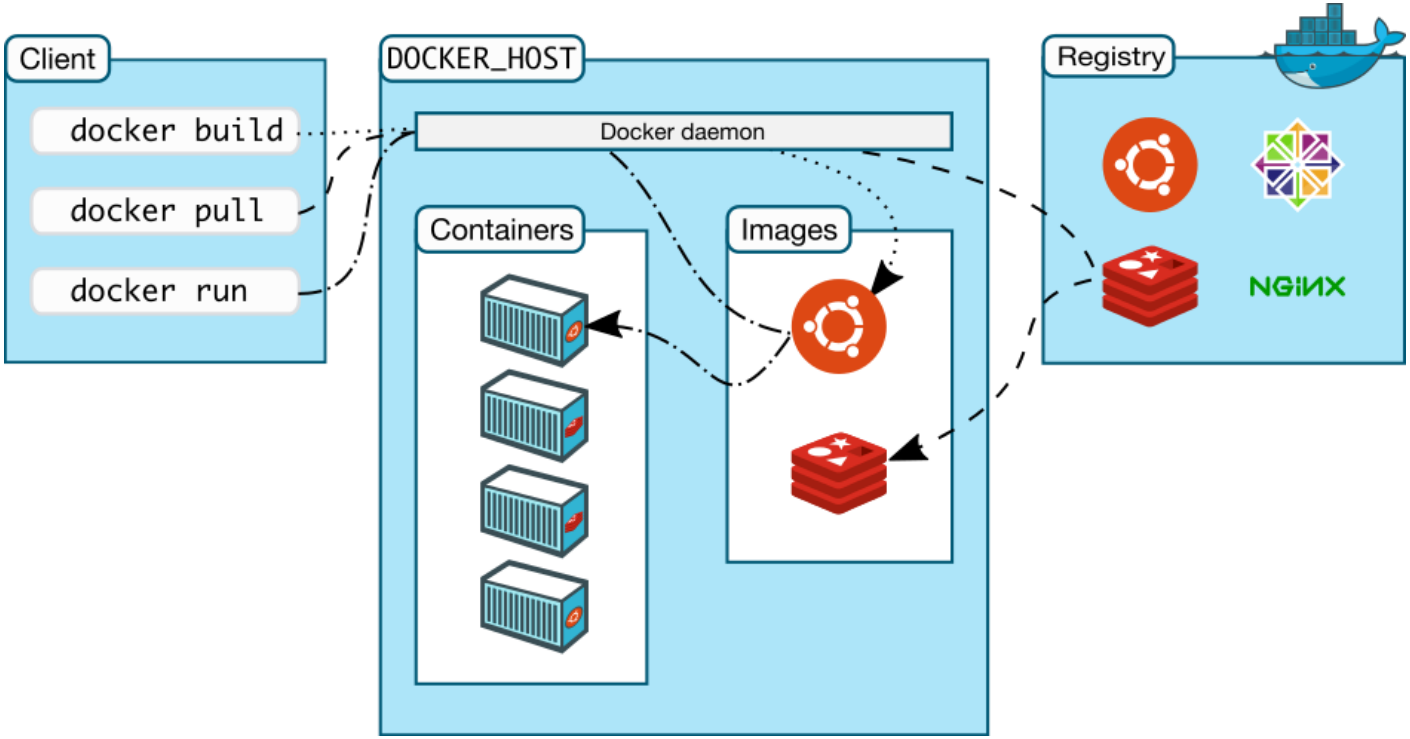
Operations

- care about containers
- work with containers
  - logging & monitoring
  - networking
  - scaling

docker

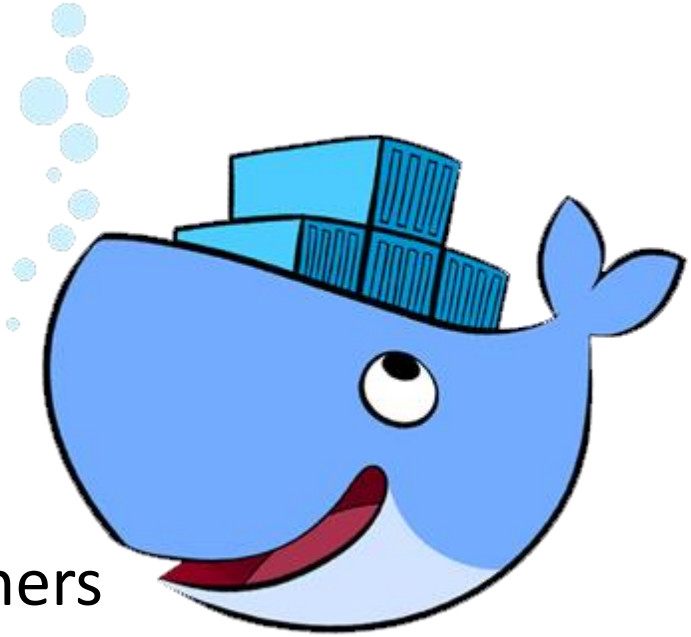
"ecosystem"

Client-Server model



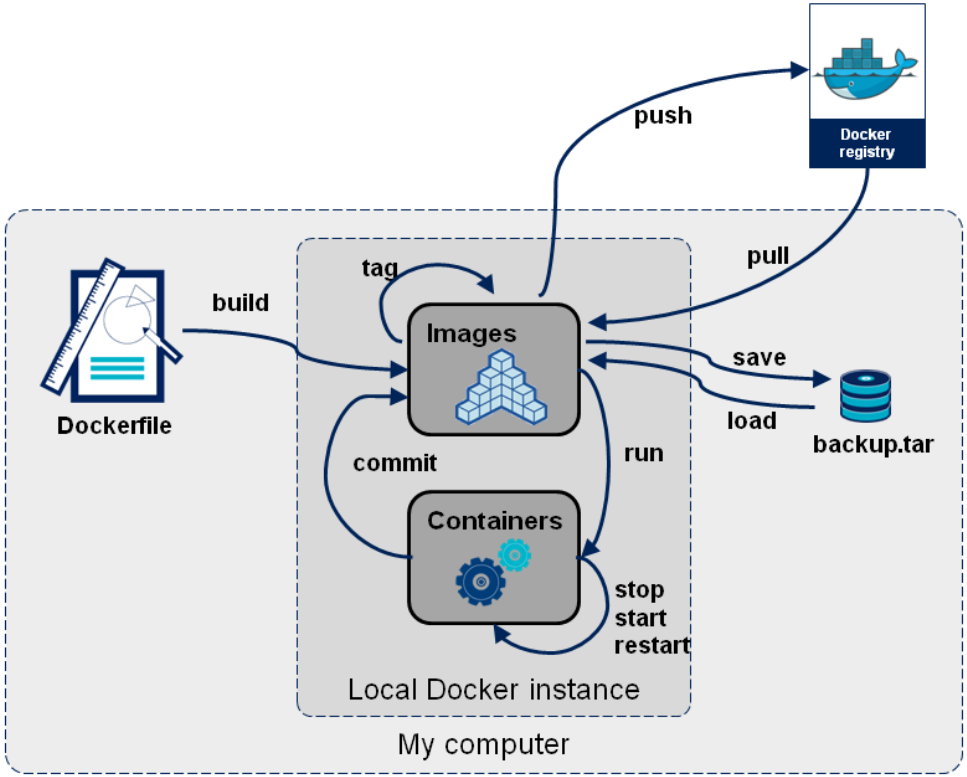
Docker engine

- runs and commoditizes Linux containers
- uses copy-on-write for quick provisioning
- runs as daemon and has CLI
- allows building & sharing images
- functionality is exposed via REST API
- defines standard format for containers



Docker commands

- ~40 commands
- work with:
  - images
  - containers
  - registry



Dockerfile-s

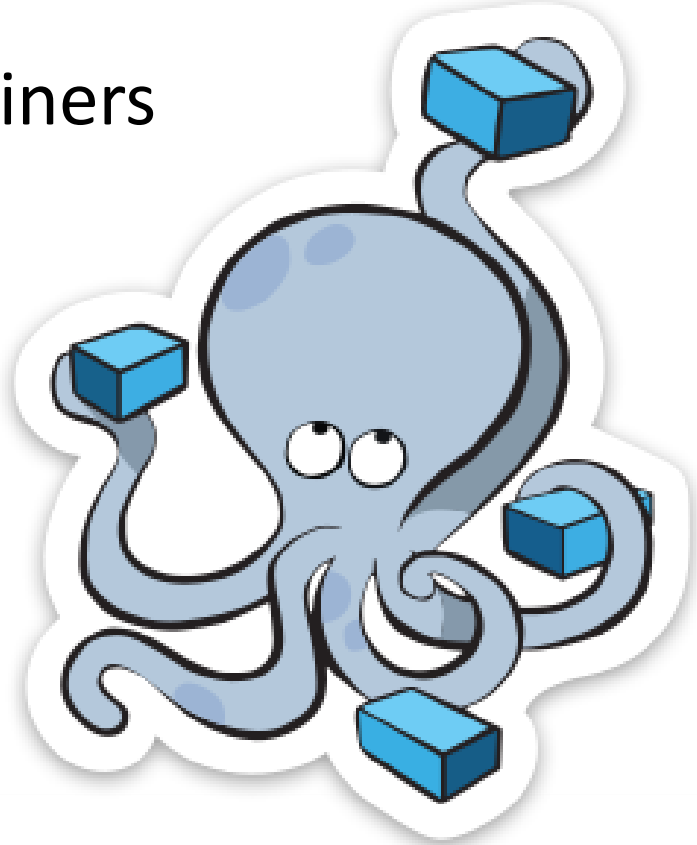
```
FROM jenkins:1.625.1
MAINTAINER Petyo Dimitrov
ENV REFRESHED_AT 2015-10-24

RUN curl -L https://github.com/... > /usr/local/bin/docker-compose
    && chmod +x /usr/local/bin/docker-compose
RUN mkdir /var/cache/jenkins
RUN chown jenkins:jenkins /var/cache/jenkins

# Set Defaults
ENV JAVA_OPTS="-Xmx6144m"
ENV JENKINS_OPTS="--handlerCountMax=300 --webroot=/var/cache/jenkins/war"

COPY plugins.txt /plugins.txt
RUN /usr/local/bin/plugins.sh /plugins.txt
```

docker build -t myjenkins .

Docker Compose

- manages a collection of containers
- fast, isolated development environments using Docker
- define env via YAML file
- quick and easy to start

```
docker-compose up -d
```



YAML example

```
proxy:
  build: nginx/
  ports:
   - "80:80"
  links:
   - appinstance
  hostname: "proxy"

nosqldb:
  build: mongo/
  hostname: "nosqldb"
  volumes:
   - "/opt/mongodb:/data/db"
```

```
appinstance:
  build: tomcat/
  expose:
   - "8080"
  ports:
   - "8180:8080"
  links:
   - nosqldb
  hostname: "appinstance"
```

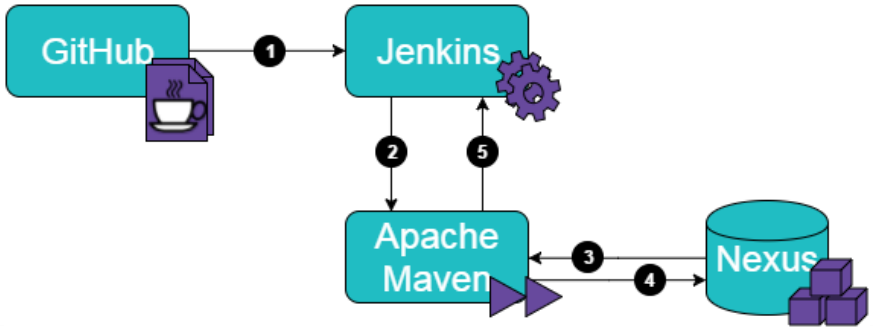
Demonstration

https://github.com/petyodimitrov/spring-music.git

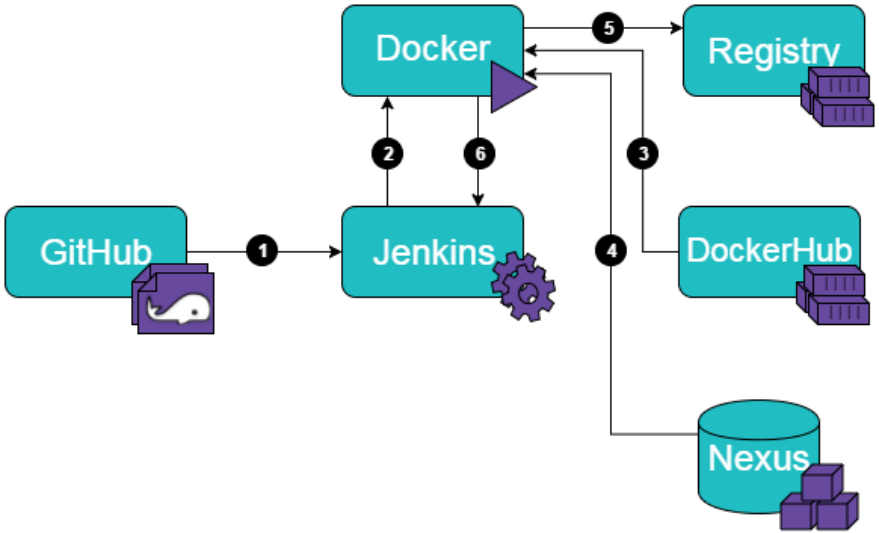
https://github.com/petyodimitrov/app-setup.git

https://github.com/petyodimitrov/ci-setup.git

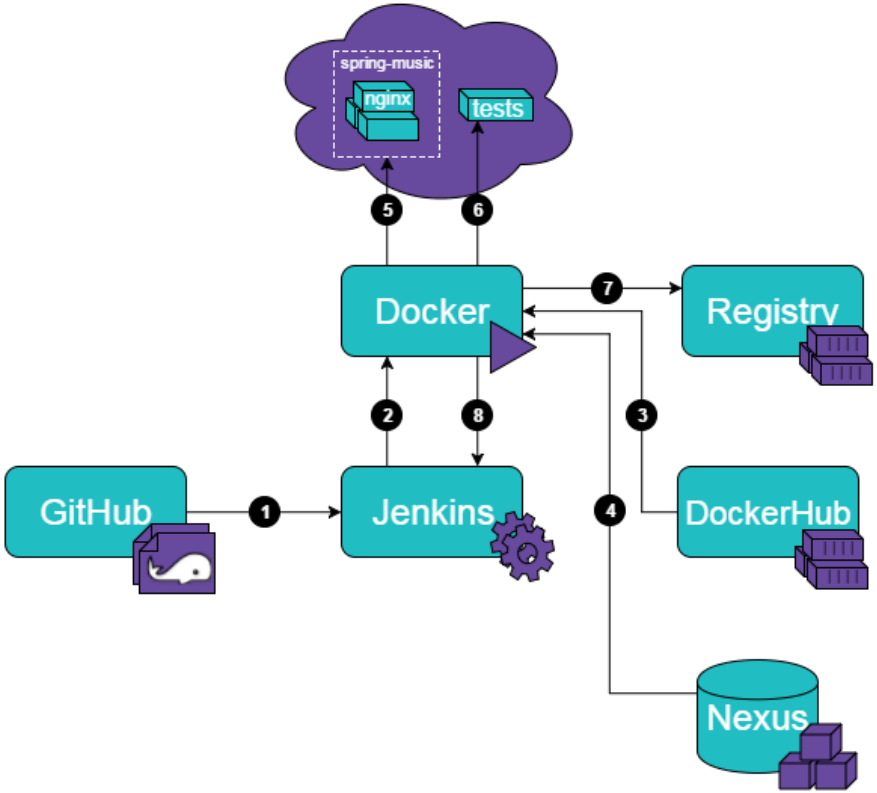
Standard Java application build



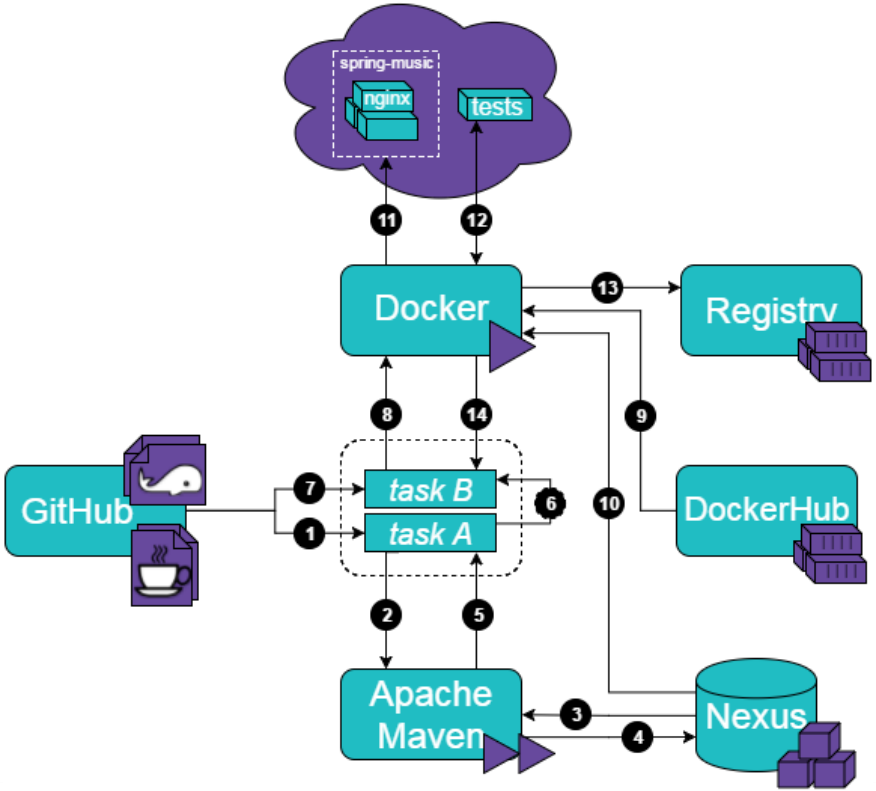
Docker containers build



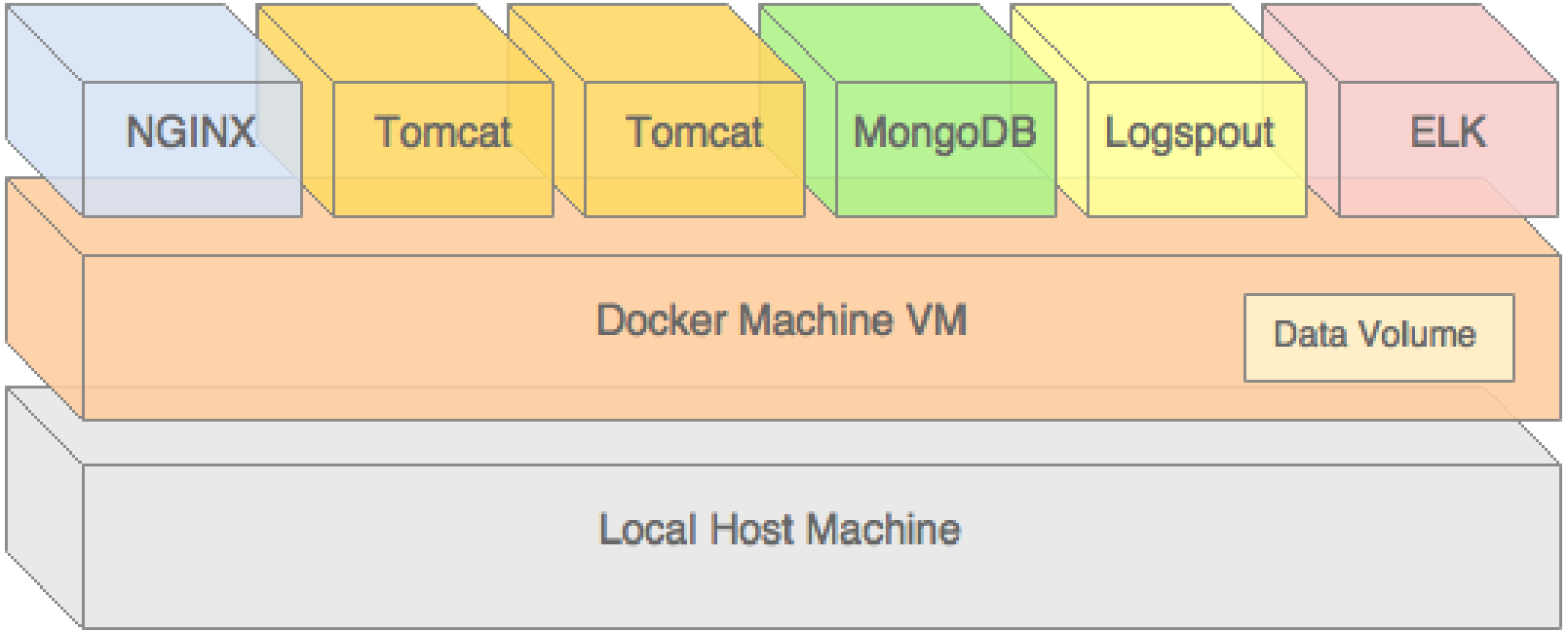
Adding system testing



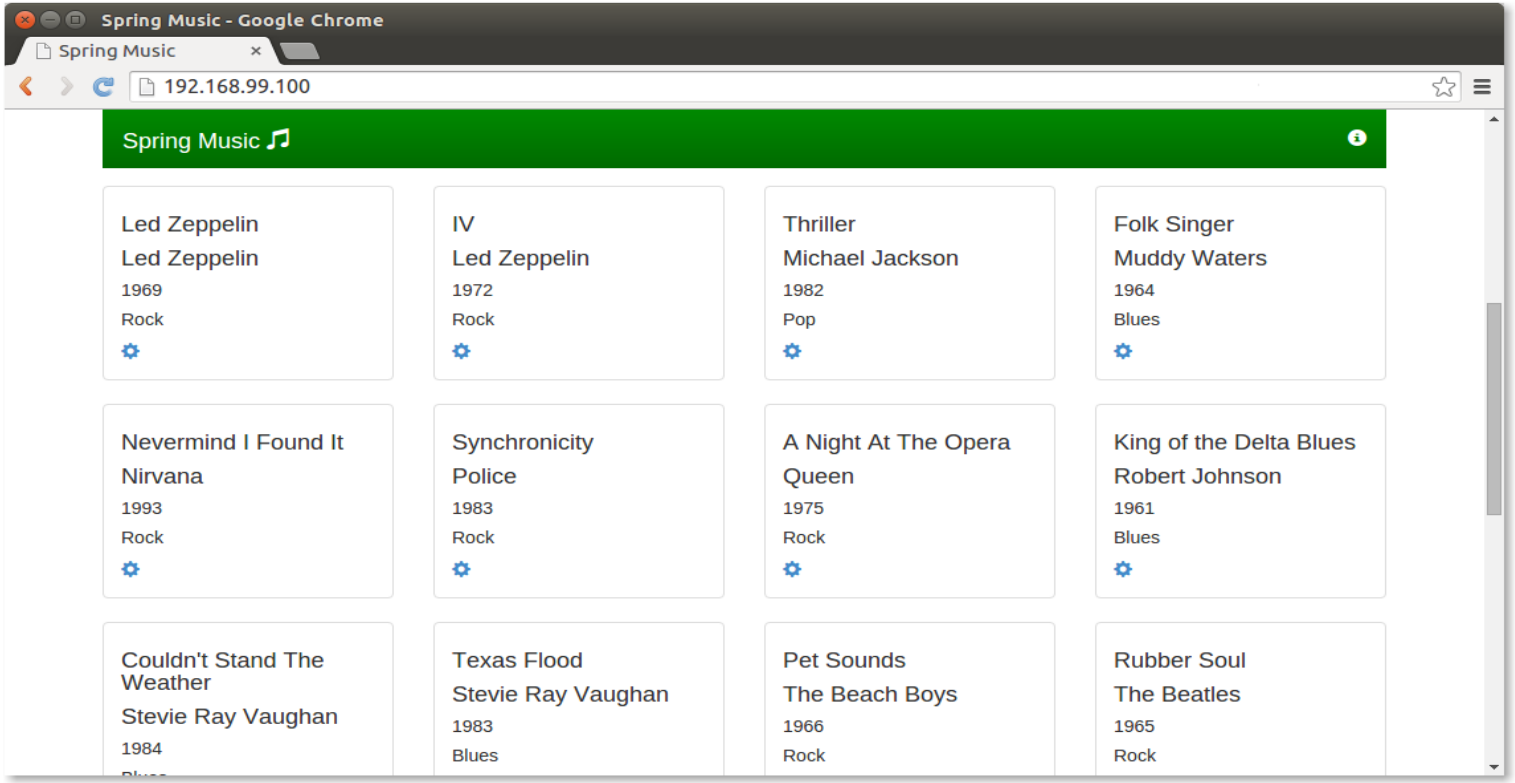
Combining all the parts



Runtime view of containers

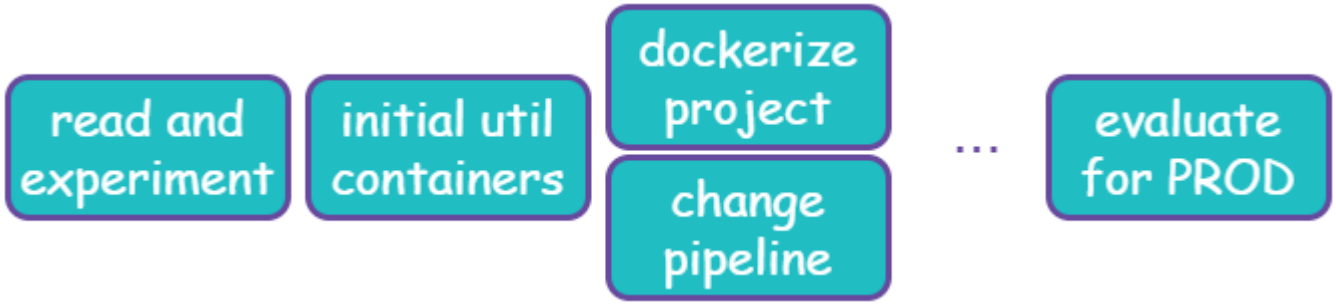


Demonstration



Tips & tricks (1)

- Decide on a strategy for introducing Docker in the project



- Keep and eye for improvements & changes (e.g. networks)
- Keep containers simple (i.e. only one/few processes)
- Automate, automate, automate … (via docker or other CM tools)
- Use initialization & validation scripts (e.g. init, supervisord)

Tips & tricks (2)

- Review Dockerfile-s as part of development process (e.g. peer reviews)

```
FROM alpine-java:14.04.3
ENV REPO=10.0.1.1:1236
RUN apt-get update -y && \
    apt-get install curl
# unnecessary
RUN curl -L https://${REPO}/something.jar
USER nobody
CMD java -jar something.jar
```

verify checksum

Q&A

petyo.dimitrov
@musala.com

MUFFIN CONFERENCE
MusalaSoft