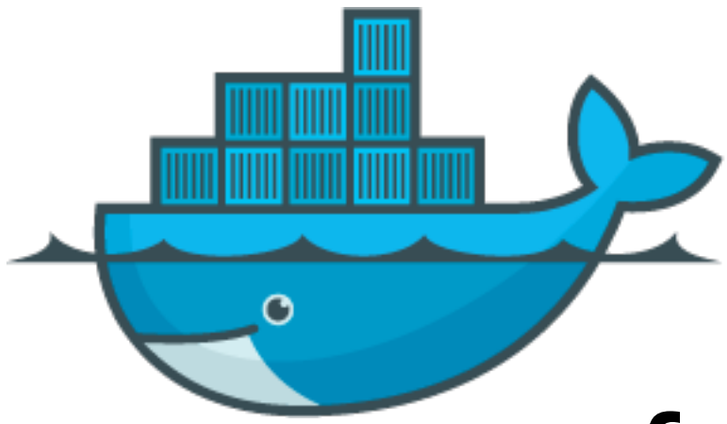


docker for Devs

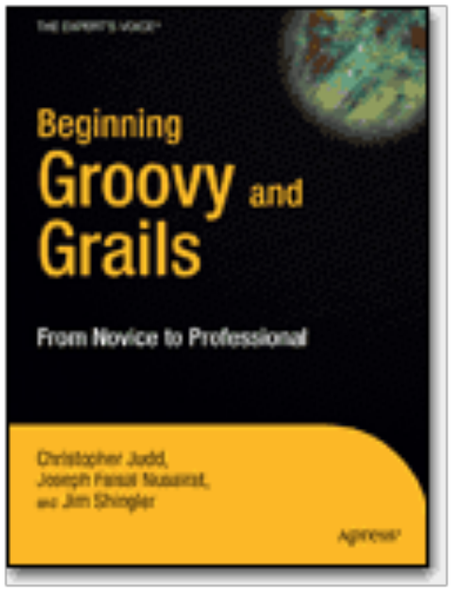
Christopher M. Judd

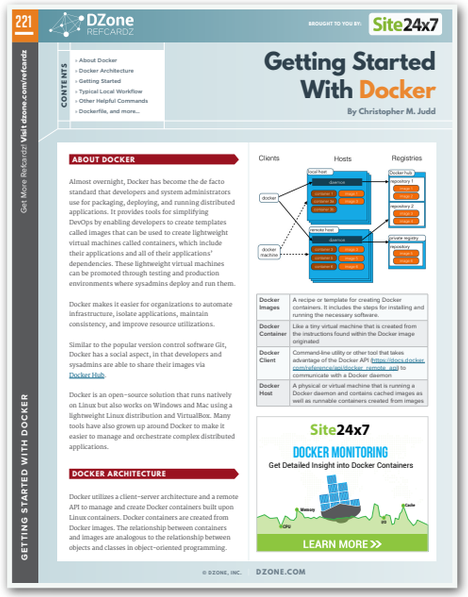
Christopher M. Judd

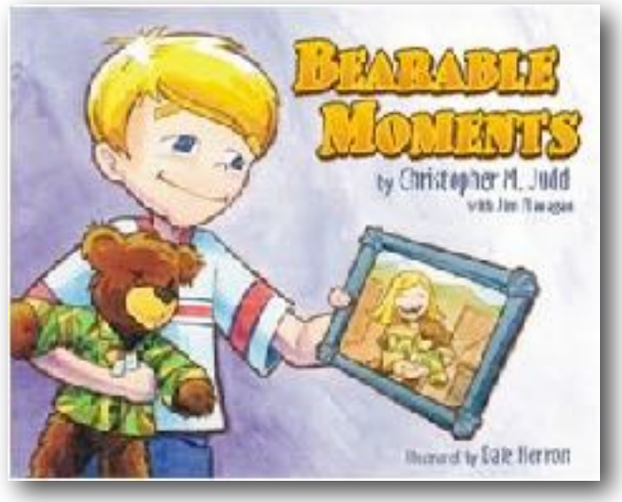
CTO and Partner at Manifest Solutions

Central Ohio Java Users Group leader

Java Champions

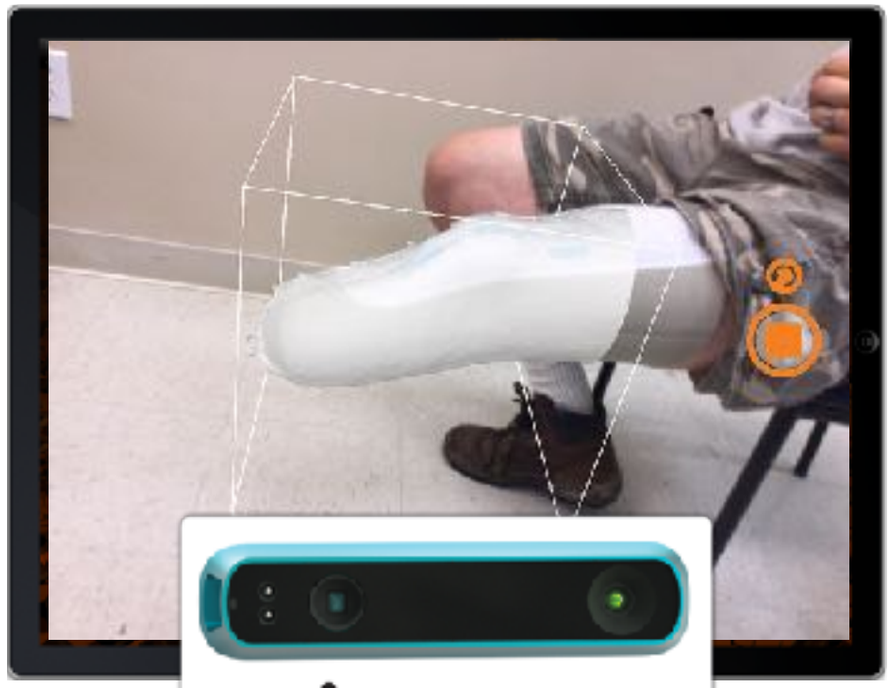




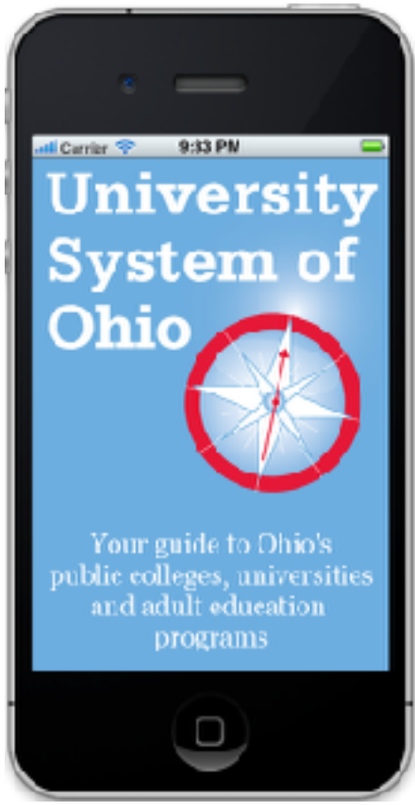



Carrier
9:32 PM
Remarkable
Ohio
Ohio | eTech Ohio
SITE OF OHIO'S FIRST STATEHOUSE


The Tabernacle
Pillar of Fire
Next




STRUCTURE
SENSOR


Carrier
9:33 PM
University System of Ohio
Your guide to Ohio's public colleges, universities and adult education programs


Carrier
12:20 PM
GERMAN VILLAGE SOCIETY
Welcome to historic German Village!
Shop
Dine
Stroll
Stay
Events
Donate


Pick a Subject
General Facts
Psychology & Philosophy
Religion
Civilization
History
Geography
Social Sciences
Political Science
Law
Education
Music

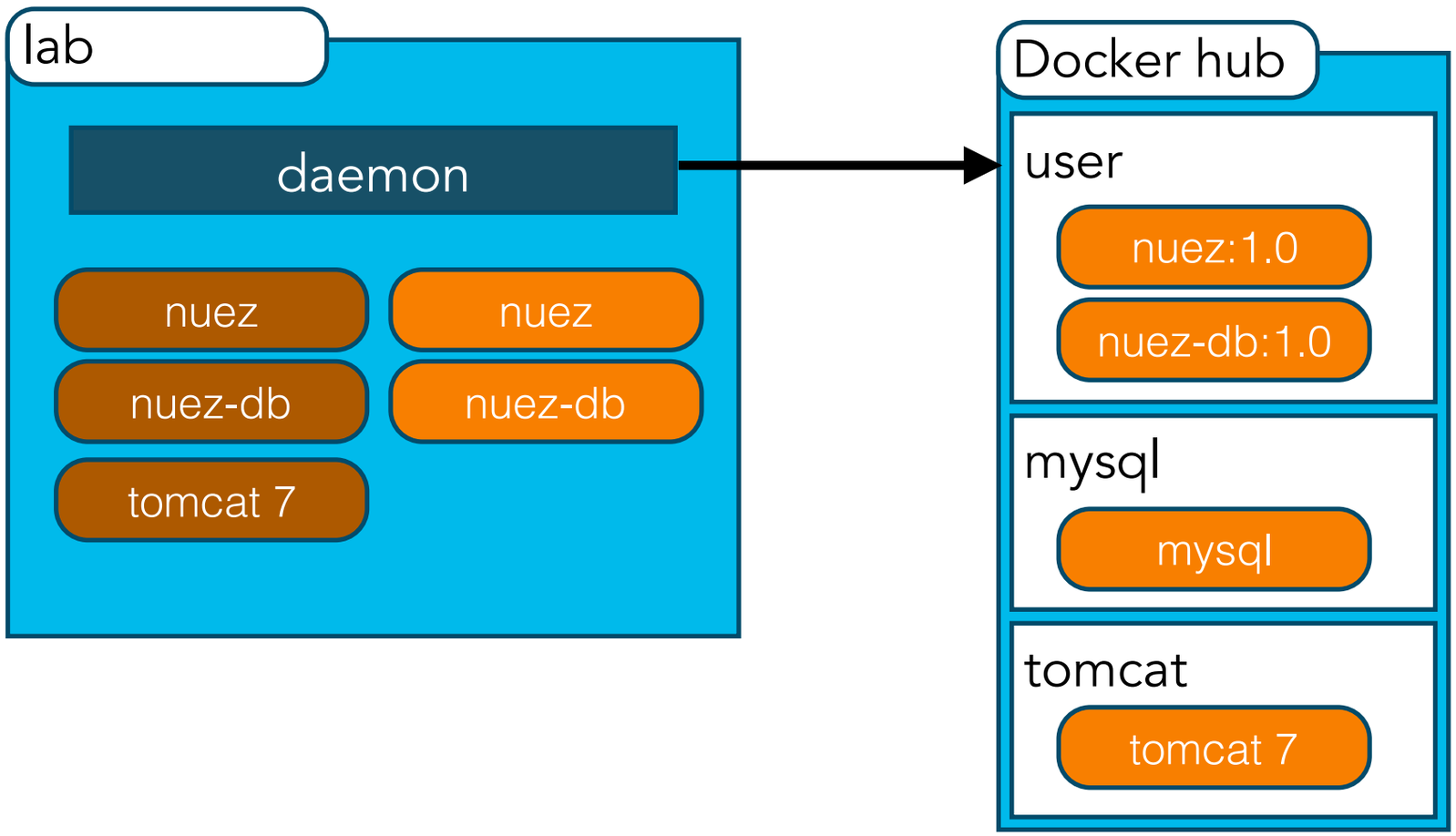

Zaner-Bloser
Handwriting
ABC
abc
123
Help
ZB
© Copyright Zaner Bloser 2010


MillerMedical
LIMBLOGIC
Active
Standby
16
18
Vacuum Level
Right


LIMBLOGIC


Ohio
PocketBallot


Carrier
2:21 PM
imSafe
mobile security worldwide


lab
daemon
nuez
nuez
nuez-db
nuez-db
tomcat 7
Docker hub
user
nuez:1.0
nuez-db:1.0
mysql
mysql
tomcat
tomcat 7

Nuez | Home

192.168.99.101

aws regatta iqity manifest willowwood judd codemash hadoop devtools devops old clients

NUEZ

The blog about anything....really...Anything!

Home All Posts About Login

Ads

Buy Stuff Here

And more stuff here

If you like stuff, you'll like this stuff...

More stuff here.

But I spent all my money on stuff.

Hello!

Welcome to the Nuez blog. Please feel free to login and blog about any topic that you want to talk about. Enjoy!

Learn more »

Search GitHub Pull requests Issues Gist

Christopher M. Judd

cjudd

Judd Solutions
Columbus Ohio
github@juddsolutions.com
http://juddsolutions.blogspot.c...
Joined on Jan 3, 2009

32 Followers **16** Starred **0** Following

Organizations

Contributions Repositories Public activity Edit profile

nuez Search All Public Private Sources Forks Mirrors New

nuez-compose

★0 ⑂0

Docker composition for the nuez-db and nuez web application.

Updated 44 minutes ago

nuez-docker

★0 ⑂0

An example of a docker file that creates an image for nuez from a URL.

Updated 20 hours ago

nuez

Groovy ★1 ⑂3

Forked from zendem/nuez

Updated 21 hours ago

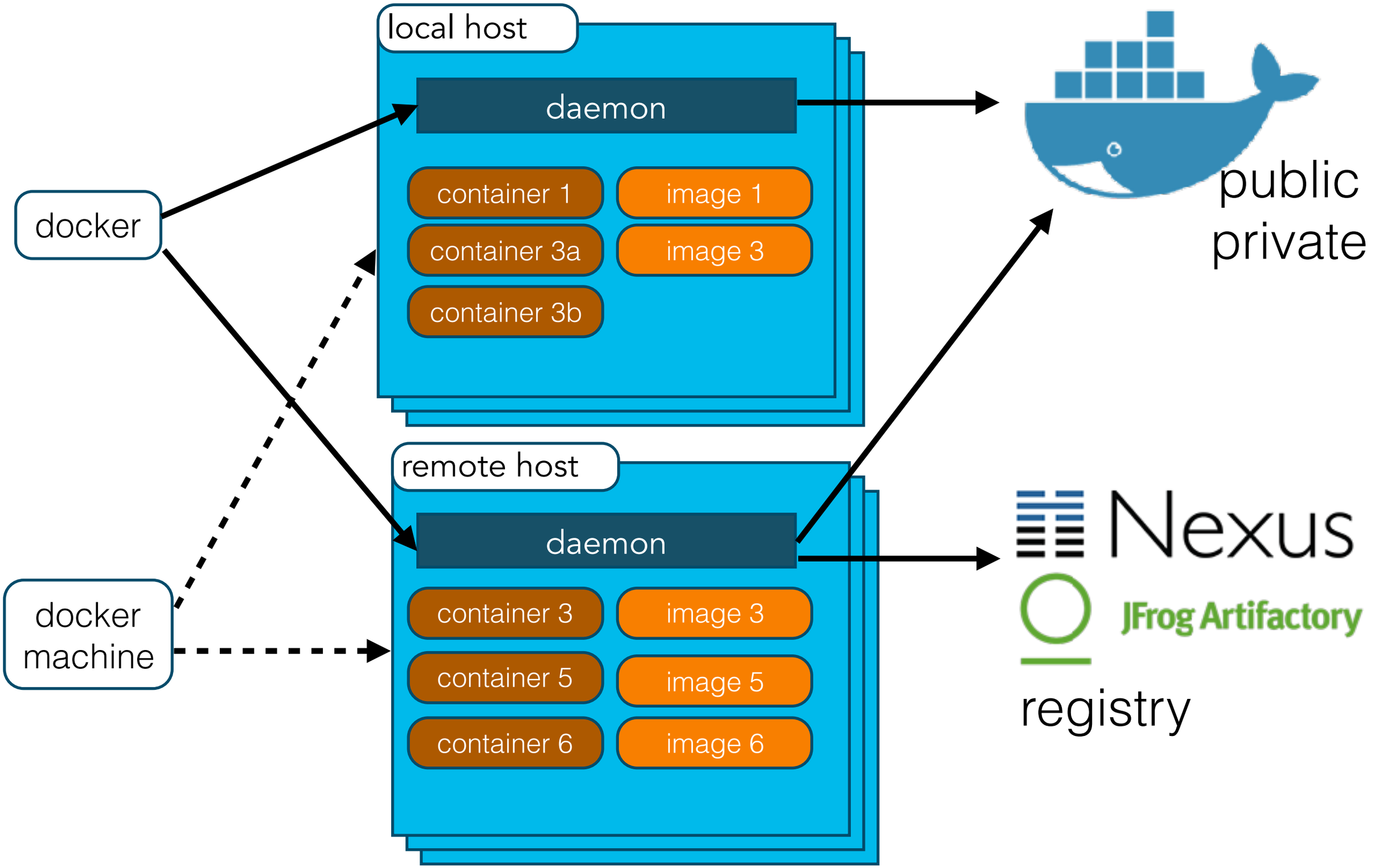
nuez-db

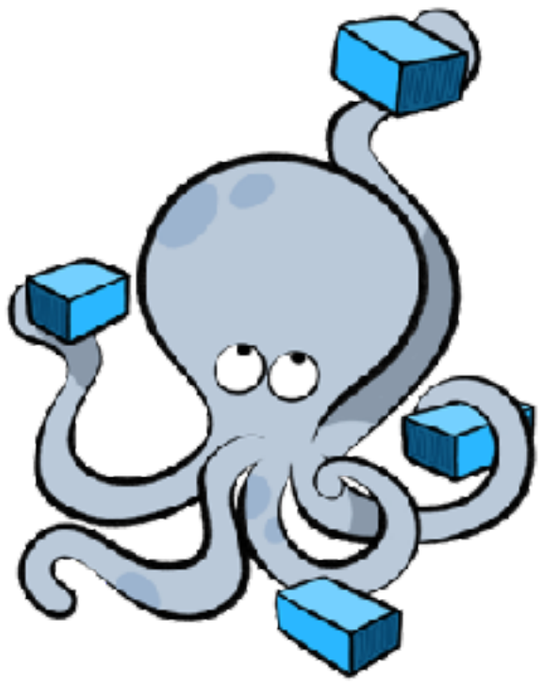
★0 ⑂0

Docker configuration for nuez-db

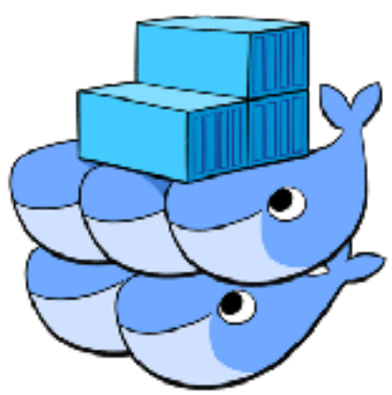
Updated a day ago

https://github.com/cjudd/nuez
https://github.com/cjudd/nuez-db
https://github.com/cjudd/nuez-docker
https://github.com/cjudd/nuez-compose


local host
daemon
container 1
image 1
container 3a
image 3
container 3b
docker
public
private
remote host
daemon
container 3
image 3
container 5
image 5
container 6
image 6
docker
machine
Nexus
JFrog Artifactory
registry



Composing
Docker API
Clustering



DigitalOcean

amazon web services™

https://goo.gl/whH1sb

Getting Started With Docker

BY **CHRISTOPHER JUDD**

CONTENTS

- About Docker
- Docker Architecture
- Other Helpful Commands
- Docker Machine
- Enterprise Docker

ABOUT DOCKER

Almost overnight, Docker has become the de facto standard that developers and system administrators use for packaging, deploying, and running distributed and cloud native applications. It provides tools for simplifying DevOps by enabling developers to create templates called images that can be used to create lightweight virtual machines called containers, which include their applications and all of their applications' dependencies. These lightweight virtual machines can be promoted through testing and production environments where sysadmins deploy and run them.

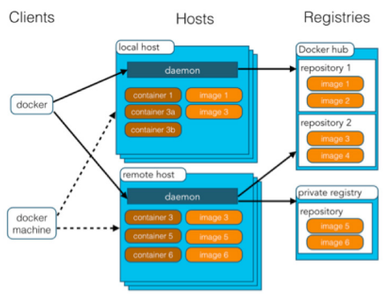
Docker makes it easier for organizations to automate infrastructure, isolate applications, maintain consistency, and improve resource utilizations.

Like the popular version control software Git, Docker has a social aspect in that developers and sysadmins can share their images via Docker Hub.

Docker is an open source solution that is available as the free Community Edition or the subscription based Enterprise Edition for multiple platforms. Docker runs natively on Linux since Docker was originally built on Linux containers but it also works on Mac and Windows. Many enterprise tools have also grown up around Docker to make it easier to manage and orchestrate complex distributed and clustered applications architectures.

DOCKER ARCHITECTURE

Docker utilizes a client-server architecture and a remote API to manage and create Docker containers and images. Docker containers are created from Docker images. The relationship between containers and images are analogous to the relationship between objects and classes in object-oriented programming, where the image describes the container and the container is a running instance of the image.



Docker Images	A recipe or template for creating Docker containers. It includes the steps for installing and running the necessary software
Docker Container	Like a tiny virtual machine that is created from the instructions found within the Docker image
Docker Client	Command-line utility or other tool that takes advantage of the Docker API (docs.docker.com/reference/api/docker_remote_api) to communicate with a Docker daemon
Docker Host	A physical or virtual machine that is running a Docker daemon and contains cached images as well as runnable containers created from images
Docker Registry	A repository of Docker images that can be used to create Docker containers. Docker Hub (hub.docker.com) is the most popular social example of a Docker repository.
Docker Machine	A utility for managing multiple Docker hosts, which can run locally in VirtualBox or remotely in a cloud hosting service such as Amazon Web Services, Microsoft Azure, Google Cloud Platform, or Digital Ocean.

https://dzone.com/refcardz/getting-started-with-docker-1

Getting Started With

Kubernetes

UPDATED BY **CHRIS JUDD**
ORIGINAL BY **ARUN GUPTA**

CONTENTS

- What Is Kubernetes?
- Key Concepts of Kubernetes
- Kubernetes Architecture
- Getting Started With Kubernetes
- Run Your First Container
- Scale Applications

WHAT IS KUBERNETES?

Kubernetes (kubernetes.io) is an open source orchestration system for managing containerized applications across multiple hosts, providing basic mechanisms for the deployment, maintenance, and scaling of applications. Originally created by Google, in March of 2016 it was donated to the Cloud Native Computing Foundation (CNCF).

Kubernetes, or "k8s" or "kube" for short, allows the user to declaratively specify the desired state of a cluster using high-level primitives. For example, the user may specify that they want three instances of the Couchbase server container running. Kubernetes' self-healing mechanisms, such as auto-restarting, re-scheduling, and replicating containers then converge the actual state towards the desired state.

Kubernetes supports Docker and Rocket containers. An abstraction around the containerization layer will allow for other container image formats and runtimes to be supported in the future.

KEY CONCEPTS OF KUBERNETES

POD

A Pod is the smallest deployable unit that can be created, scheduled, and managed. It's a logical collection of containers that belong to an application.

Each resource in Kubernetes is defined using a configuration file. For example, a Couchbase pod can be defined with the following `.yaml` file:

```
apiVersion: v1
kind: Pod
# labels attached to this Pod
metadata:
  name: couchbase-pod
  labels:
    name: couchbase-pod
spec:
  containers:
  - name: couchbase
    # Docker image that will run in this Pod
    image: couchbase
    ports:
    - containerPort: 8091
```

LABEL

A label is a key/value pair that is attached to objects, such as pods. In the previous example, `metadata.labels` define the labels attached to the pod.

Labels define identifying attributes for the object and is only meaningful and relevant to the user. Multiple labels can be attached to a resource. Labels can be used to organize and to select subsets of objects.

REPLICA SETS

A replica set ensures that a specified number of pod replicas are running on worker nodes at any one time. It allows both up- and down-scaling the number of replicas. It also ensures recreation of a pod when the worker node reboots or otherwise fails.

NOTE: Replica Sets replaces Replication Controllers.

A Replica Set creating two instances of a Couchbase pod can be defined as:

```
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: couchbase-rs
spec:
  # Two replicas of the Pod to be created
  replicas: 2
  # Identifies the label key and value on the Pod that
  # this Replica Set is responsible for managing
  selector:
    matchLabels:
      app: couchbase-rs-pod
    matchExpressions:
      - {key: tier, operator: In, values: ["backend"]}
  template:
    metadata:
      labels:
        # label key and value on the pod.
        # These must match the selector above.
        app: couchbase-rs-pod
        tier: backend
  spec:
    containers:
    - name: couchbase
      image: couchbase
      ports:
      - containerPort: 8091
```


https://dzone.com/refcardz/kubernetes-essentials

My Journey to Enlightenment

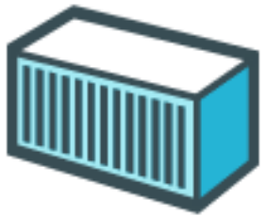
https://www.flickr.com/photos/kubina/942527814/


Start
Ivan
Skype
Reading List
Internet Explorer
Help+Tips
SkyDrive
34°
Washington D.C.
Snow
Today
34°/22° Clear
Tomorrow
31°/21° Mostly Sunny
Weather
Desktop
Photos
mint.com
New to the Store
Mint.com
Free Not yet rated
1
Football-Friendly: Chef Marcus Samuelsson's...
Maps


docker

An open platform for distributed applications for developers and sysadmins

Build



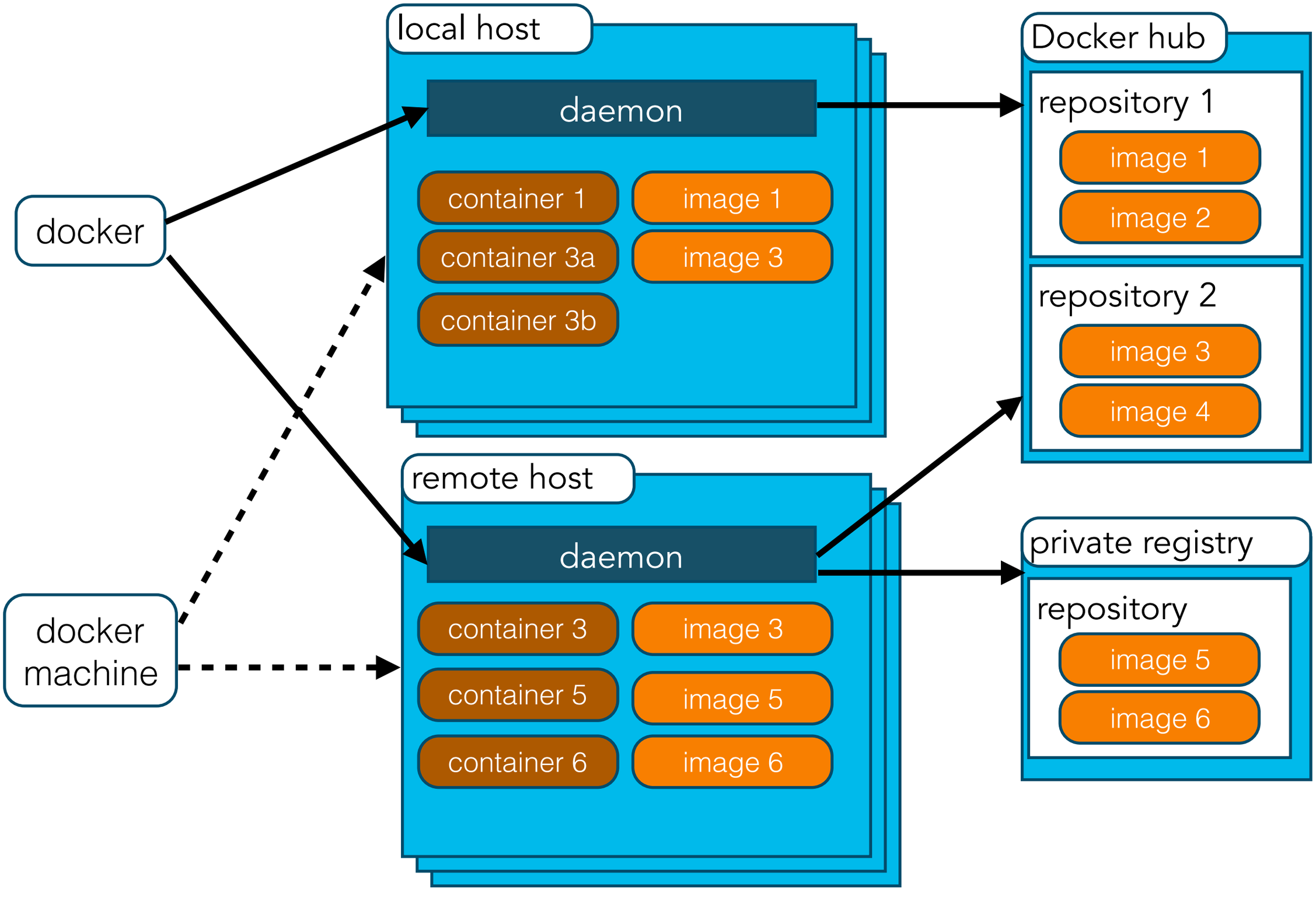
Ship

Run

Clients

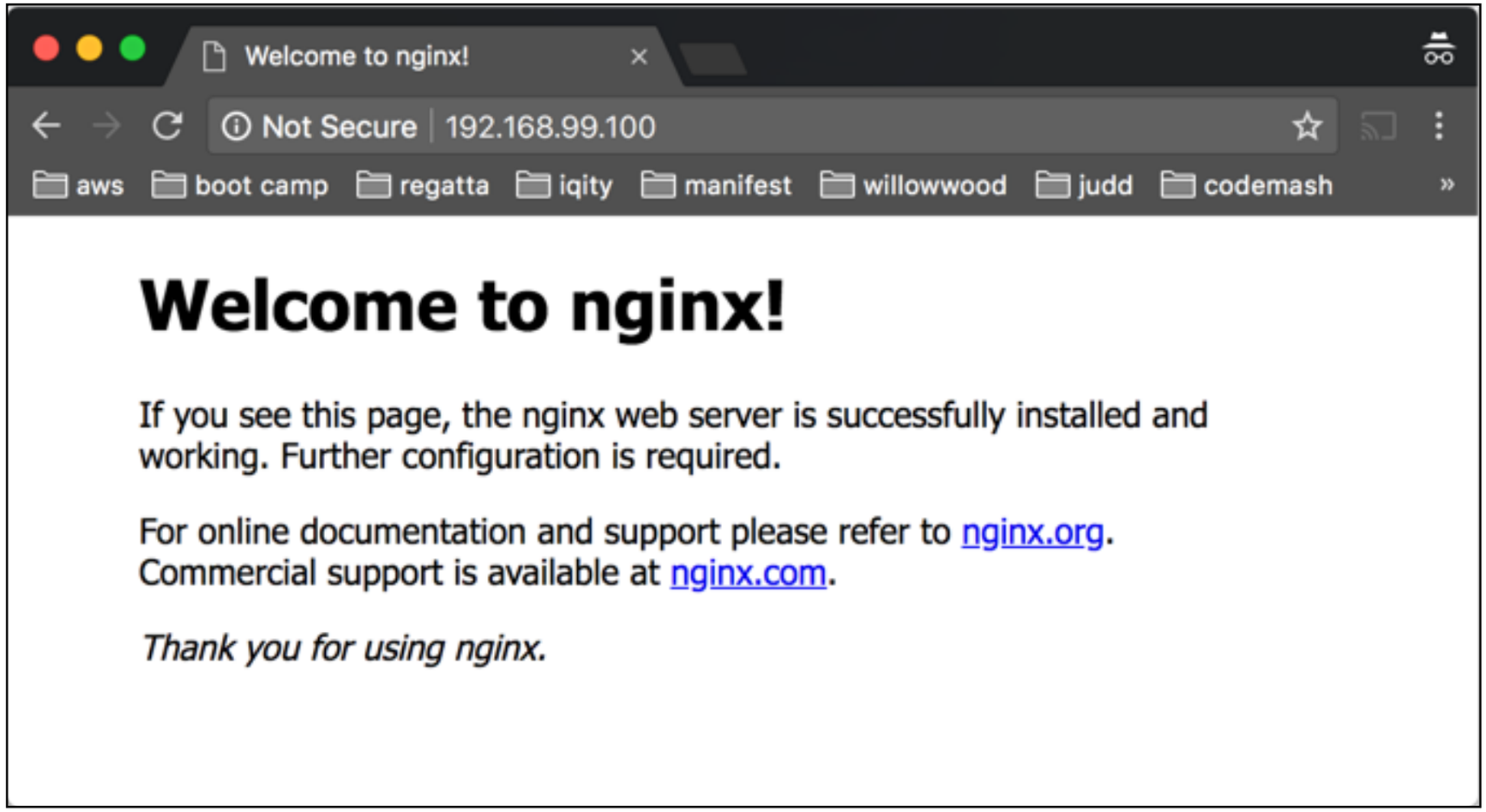
Hosts

Registries



```
docker run -it --rm -p 80:80 nginx
```

```
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
8176e34d5d92: Pull complete
5b19c1bdd74b: Pull complete
4e9f6296fa34: Pull complete
Digest: sha256:4771d09578c7c6a65299e110b3ee1c0a2592f5ea2618d23e4ffe7a4cab1ce5de
Status: Downloaded newer image for nginx:latest
```



- infrastructure automation
- sandboxing/isolation
- maintain consistency
- better resource utilization
- easy experimentation

infrastructure automation

Run the following commands

1. apt-get mysql unless you are on CentOS
2. vim /etc/timezone
   and add America/New_York
3. vim /etc/mysql/my.cnf
   and add federated
   and add lower_case_table_names = 1

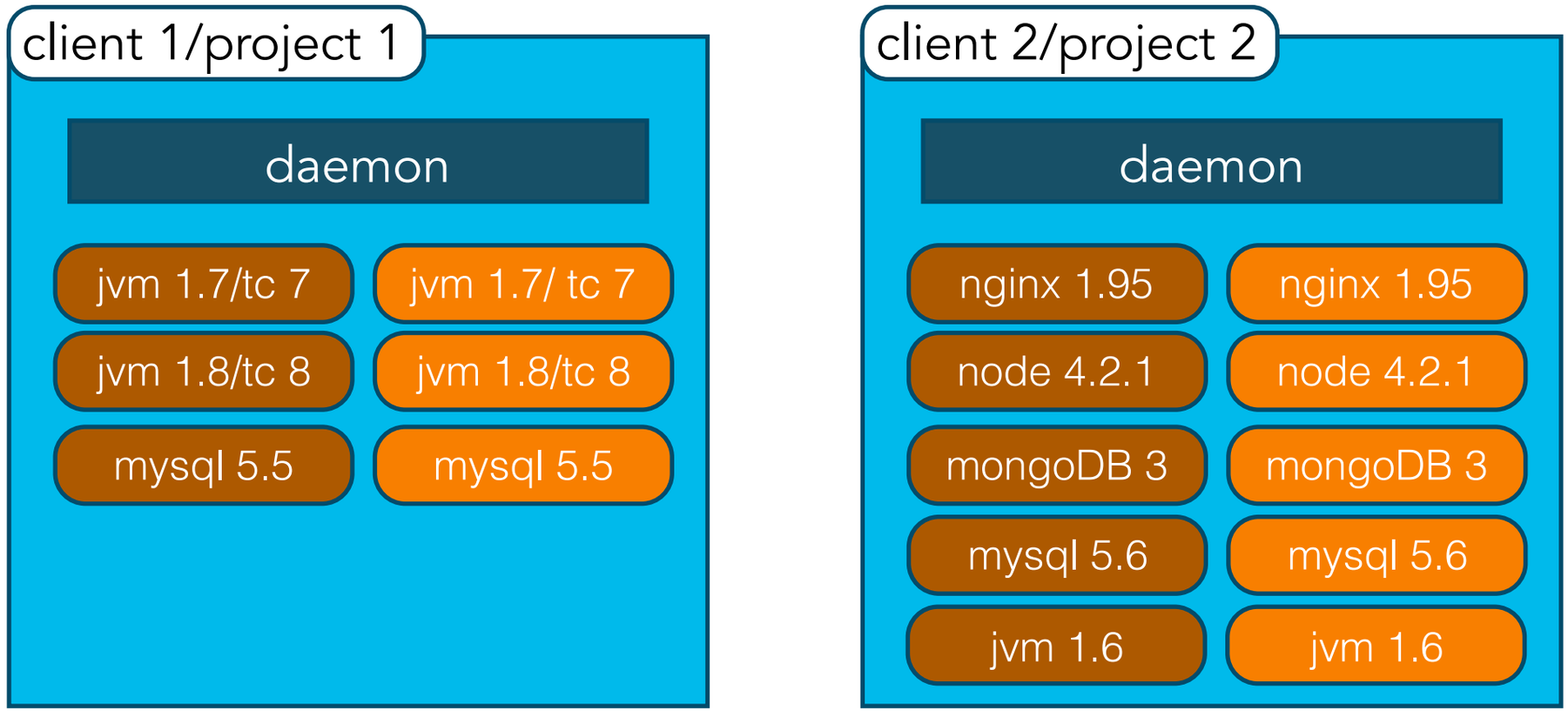
infrastructure automation

```
FROM mysql:5.5.45

COPY my-custom-entrypoint.sh /

RUN echo America/New_York | tee /etc/timezone
        && dpkg-reconfigure --frontend noninteractive tzdata
RUN echo "federated" >> /etc/mysql/my.cnf
RUN echo "lower_case_table_names = 1" >> /etc/mysql/my.cnf

ENTRYPOINT ["/my-custom-entrypoint.sh"]
CMD ["mysqld"]
```

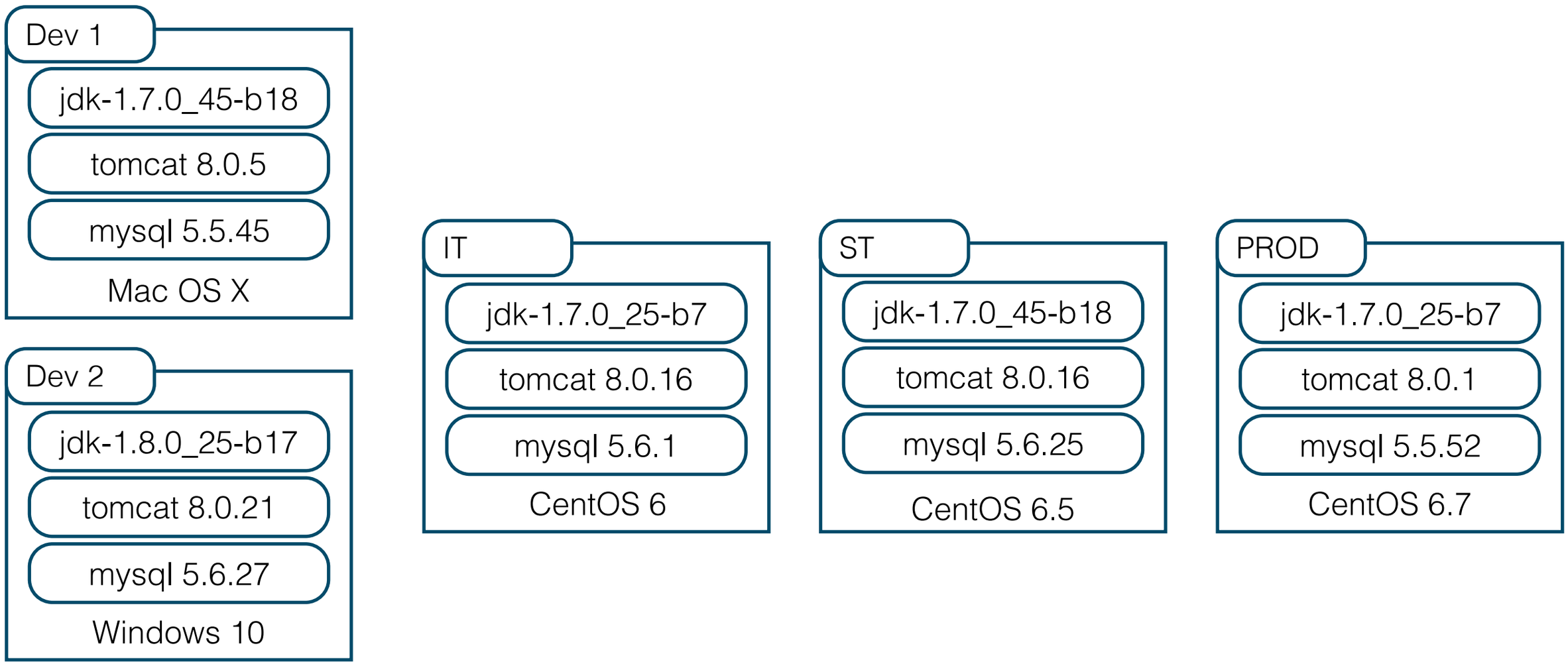
sandboxing/isolation

```
apache-maven-2.2.1
apache-maven-3.0.4
apache-maven-3.2.1
apache-tomcat-5.5.17
apache-tomcat-6.0.29
apache-tomcat-7.0.37
apache-tomcat-8.0.21
apache-tomcat-8.0.27
grails-1.3.5
grails-1.3.6
grails-1.3.7
grails-2.0.0
grails-2.1.1
grails-2.1.4
grails-2.2.1
grails-2.2.2
grails-2.3.10
groovy-1.7.10
groovy-2.0.5
groovy-2.1.1
groovy-2.2.2
jdk-1.6.0_65-b14-462
jdk-1.7.0_45-b18
jdk-1.8.0_25-b17
mysql-5.6.15
```

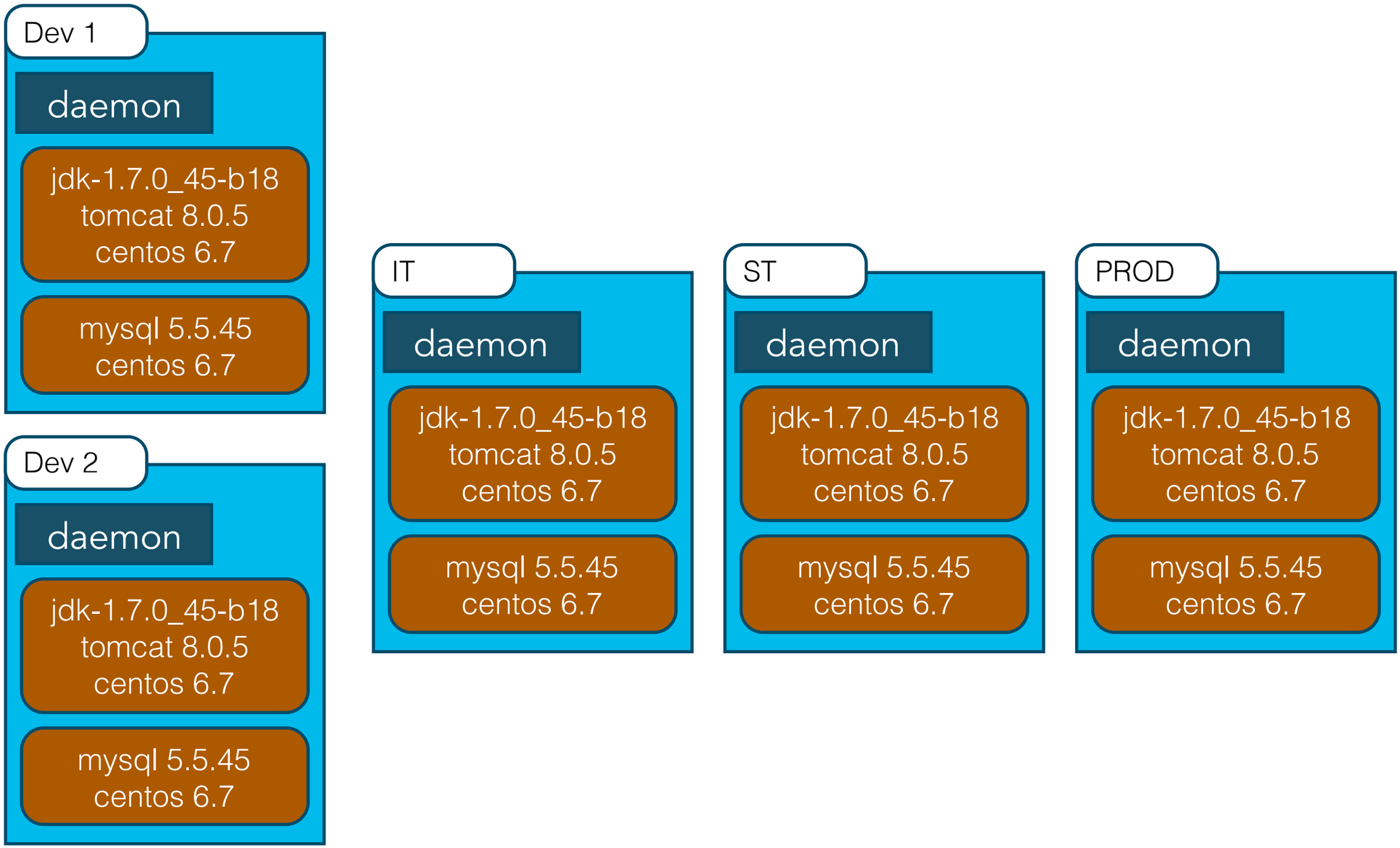
sandboxing/isolation



maintain consistency

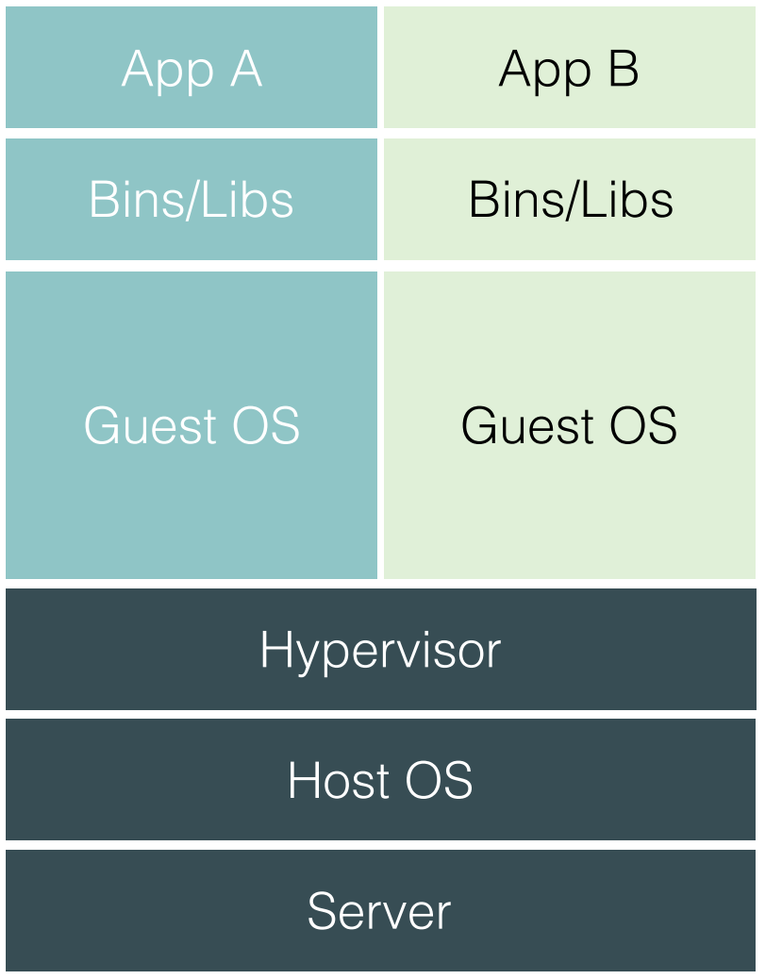


maintain consistency

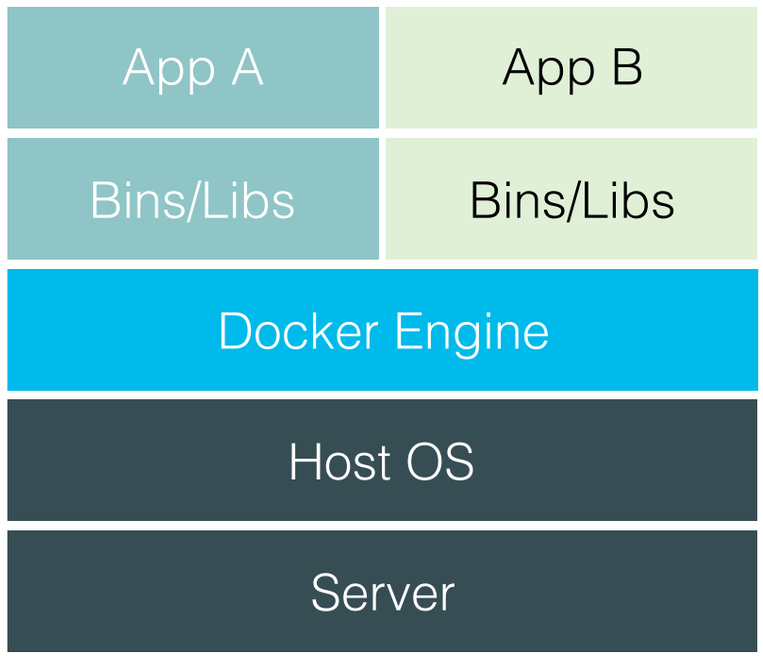


better resource utilization

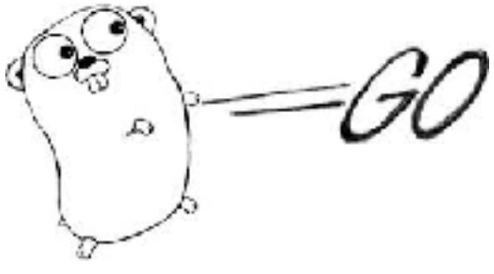
Virtual Machine



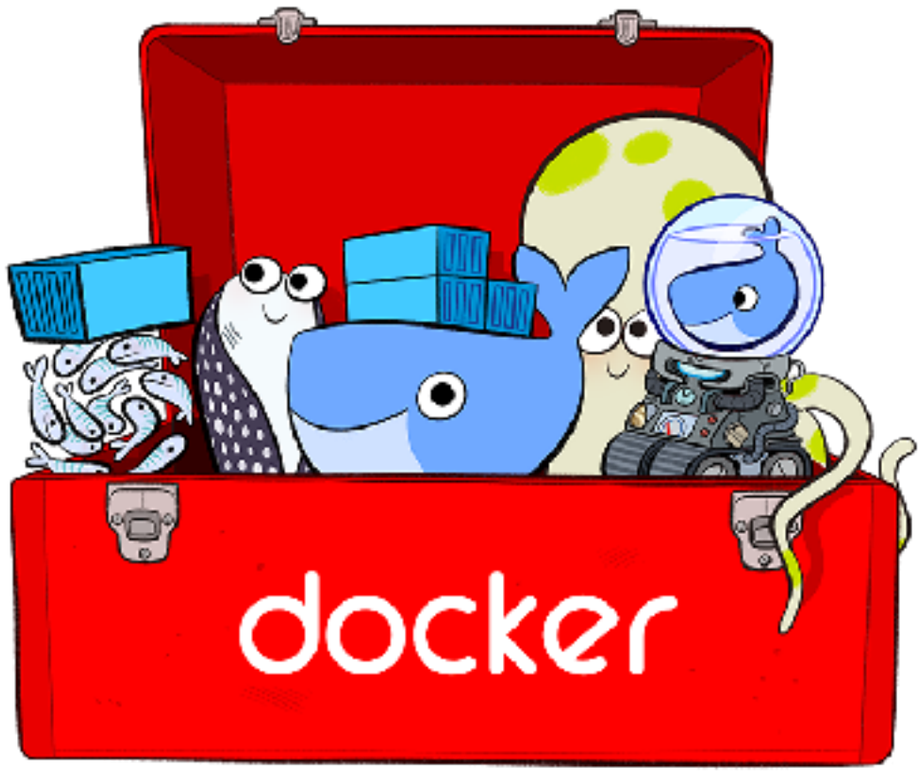
Docker



easy experimentation



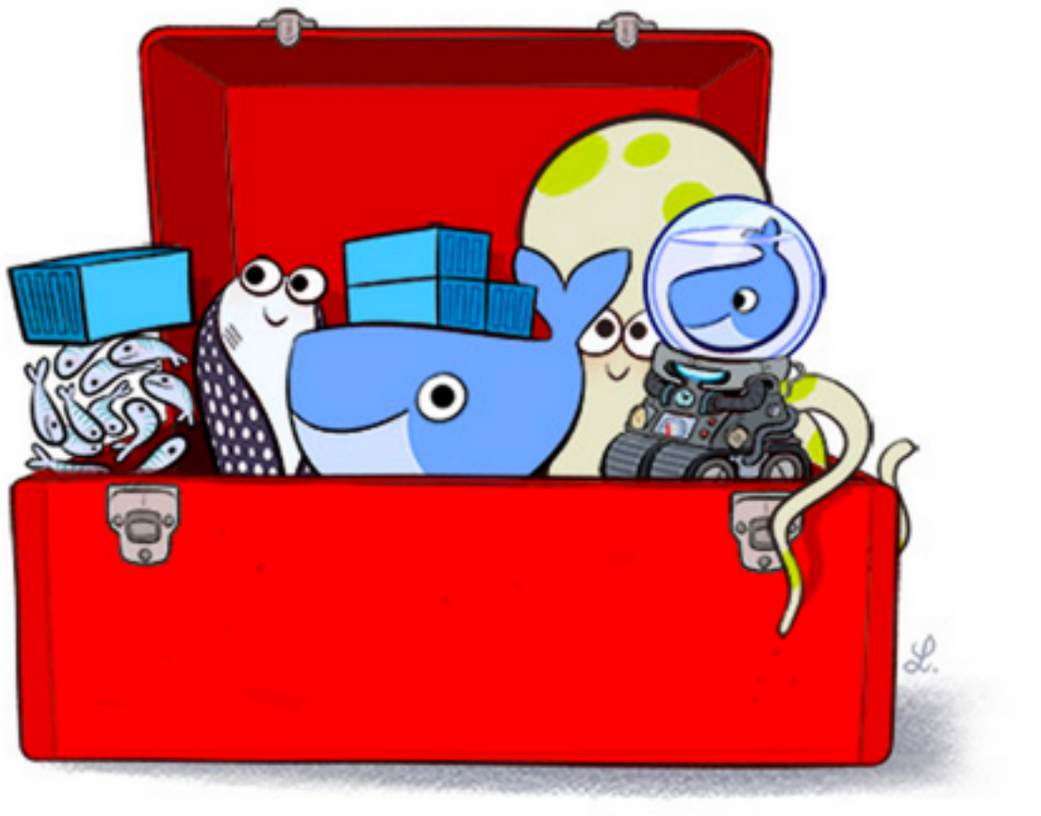
Setup Docker



VS

Docker Toolbox

Getting Started Guide (Mac) | Getting Started Guide (Windows) | Contribute to Toolbox



Download (Mac) Download (Windows)

Compatible with Mac OS X 10.8+ and Windows 7+

https://www.docker.com/toolbox

host

VirtualBox

default

daemon

container 1

image 1

IP 192.168.99.100

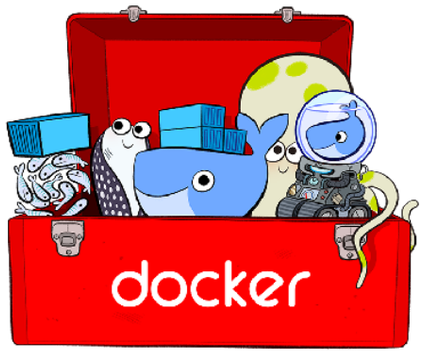
machine 2

daemon

container 2

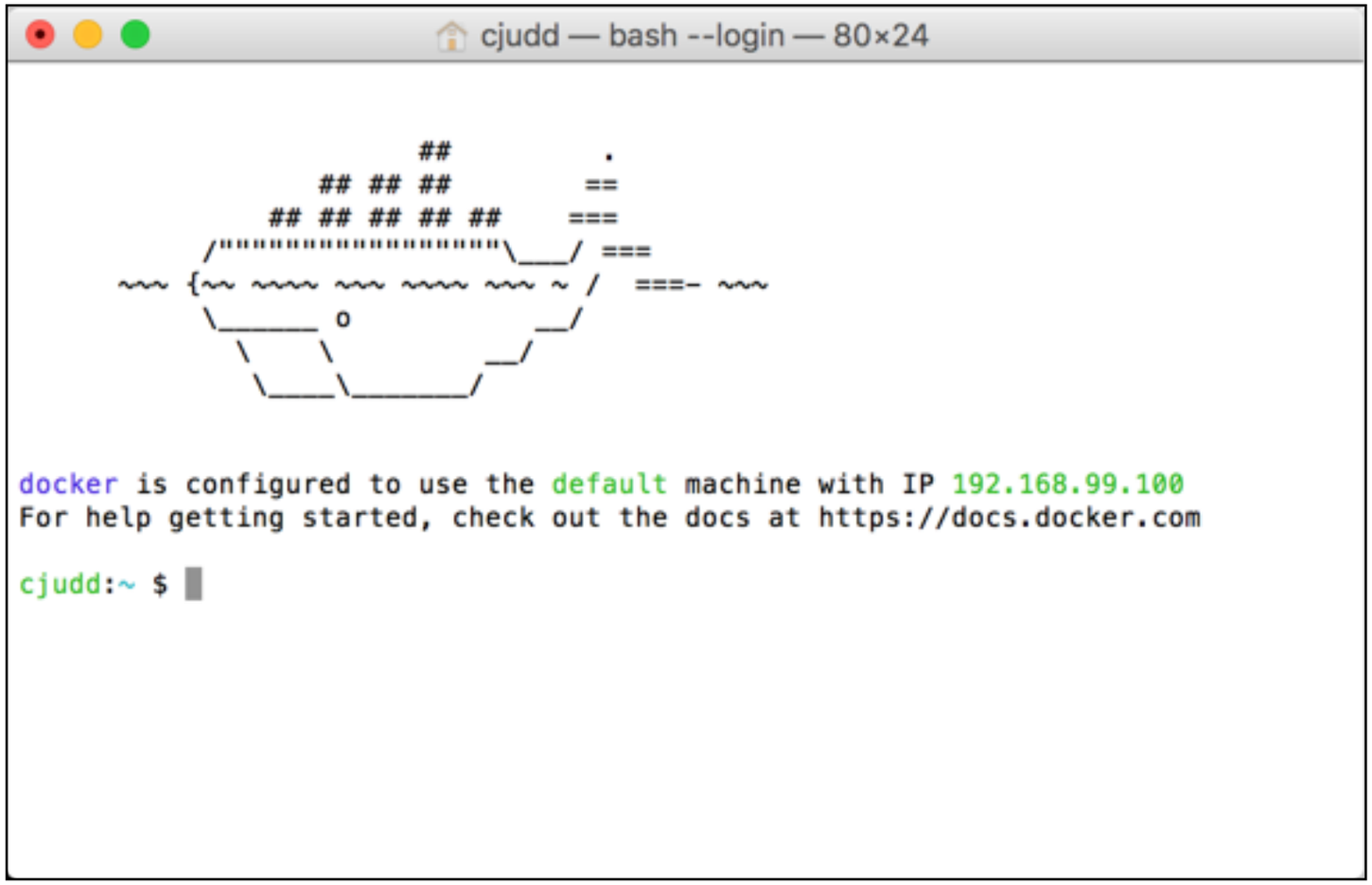
image 2

IP 192.168.99.101



- Enable Virtualization
  - https://docs.docker.com/engine/installation/windows/
- Turn Off Hyper-V
  - `bcdedit /set hypervisorlaunchtype off`
  - restart
- Install latest version of Virtual Box (5.0.12)
- Install Docker Toolbox
  - uncheck “Install Virtual Box” checkbox

```
bcdedit /set hypervisorlaunchtype auto
```



Docker Quickstart Terminal

LOGIN

Containers

+ NEW

nuez-db
nuez-db

Search for Docker images from Docker Hub

FILTER BY All Recommended My Repos My Images

Recommended

kitematic
hello-world-nginx
A light-weight nginx container that demonstrates the features of Kitematic
♡67 ⇩1M CREATE

official
ghost
Ghost is a free and open source blogging platform written in JavaScript
♡497 ⇩3M CREATE

official
jenkins
Official Jenkins Docker image
♡2.2K ⇩13M CREATE

official
redis
Redis is an open source key-value store that functions as a data structure server.
♡3.1K ⇩172M CREATE

official
rethinkdb
RethinkDB is an open-source, document database that makes it easy to build and scale real-time...
♡351 ⇩5M CREATE

kitematic
minecraft
The Minecraft multiplayer server allows two or more players to play Minecraft together
♡75 ⇩34K CREATE

official
solr
Solr is the popular, blazing-fast, open source enterprise search platform built on Apache...
♡304 ⇩664K CREATE

official
elasticsearch
Elasticsearch is a powerful open source search and analytics engine that makes data easy to...
♡1.8K ⇩31M CREATE

official
postgres
The PostgreSQL object-relational database system provides reliability and data integrity.
♡2.9K ⇩27M CREATE

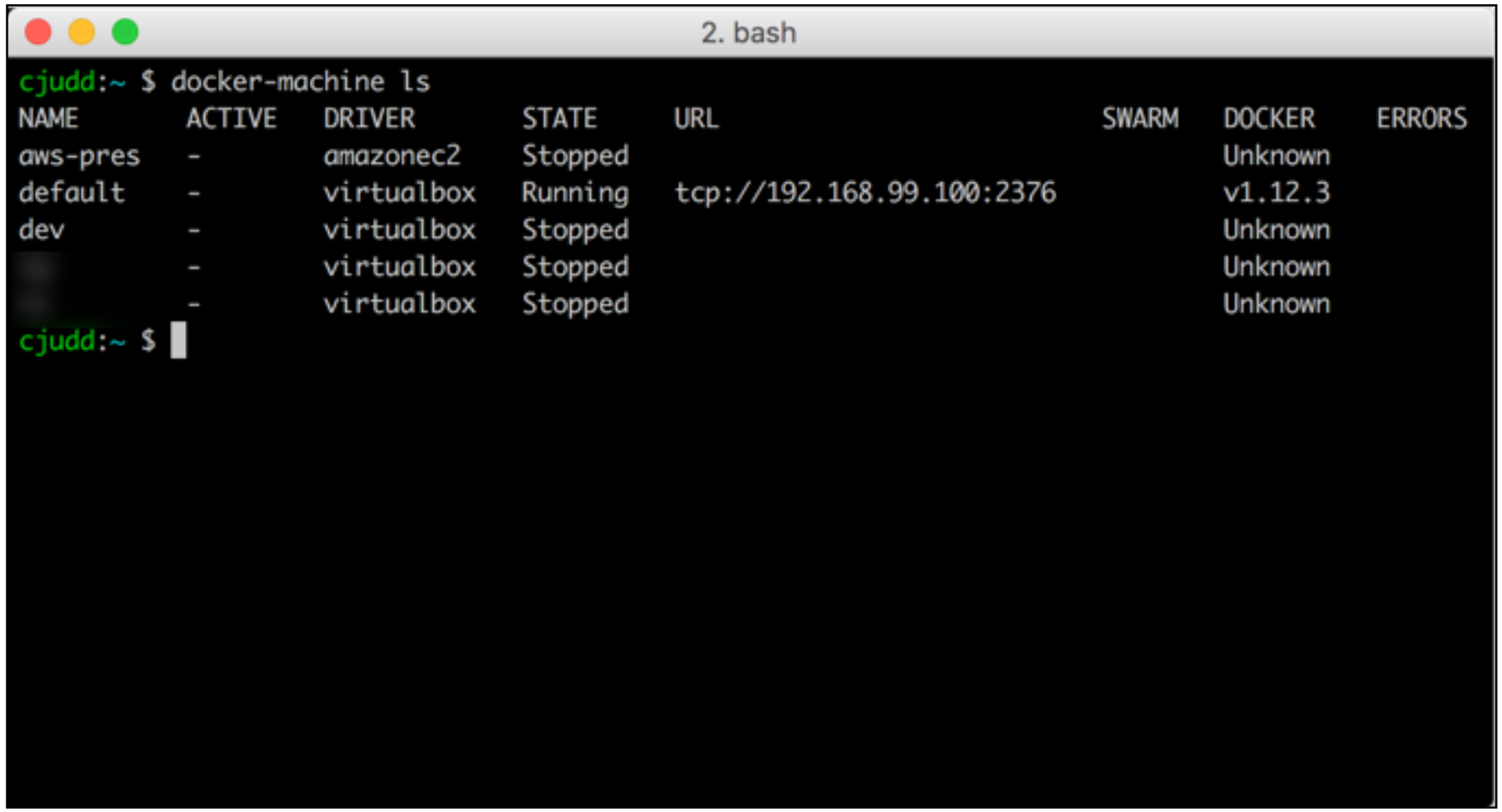
official
ubuntu-upstart
Upstart is an event-based replacement for the /sbin/init daemon which starts processes...
♡69 ⇩275K CREATE

official
memcached
Free & open source, high-performance, distributed memory object caching system.
♡623 ⇩7M CREATE

official
rabbitmq
RabbitMQ is a highly reliable enterprise messaging system based on the emerging AMQP...
♡1.1K ⇩11M CREATE

DOCKER CLI

Kitematic



command-line

docker

DOCS EVENTS COMMUNITY SUPPORT TRAINING PARTNERS BLOG LOG-IN SIGN-UP

What Is Docker? Solutions Get Docker Pricing Open Source Company

Welcome to the Docs

Docker ID

Docker Engine

Docker for Mac

- Getting Started
- Docker for Mac vs. Docker Toolbox
- Leveraging Multi-CPU Architecture Support
- Networking
- File system sharing
- Logs and Troubleshooting
- FAQs
- Example Applications
- Open Source Licensing
- Release Notes

Docker for Windows

Docker Compose

CS Docker Engine

Search the doc

Edit This Page

Get started with Docker for Mac

Download Docker for Mac

What to know before you install

Step 1. Install and Run Docker for Mac

Step 2. Check versions of Docker Engine, Compose, and Machine

Step 3. Explore the application and run examples

Preferences

General

Advanced

Docker Daemon [Beta feature]

File sharing

Privacy

Uninstall or reset

Installing bash completion

Where to go next

Get started with Docker for Mac

Estimated reading time: 16 minutes

Welcome to Docker for Mac!

Please read through these topics on how to get started. To **give us feedback** on your experience with the app and report bugs or problems, log in to our Docker for Mac forum.

> **Already have Docker for Mac?** If you already have Docker for Mac installed, and are ready to get started, skip over to the Getting Started with Docker tutorial.

Download Docker for Mac

If you have not already done so, please install Docker for Mac. You can download installers from the stable or beta channel.

For more about stable and beta channels, see the FAQs.

Stable channel

This installer is fully baked and tested, and comes with the latest GA version of Docker Engine.

This is the best channel to use if you want a reliable platform to work with.

These releases follow a version schedule with a longer lead time than the betas, synched with Docker Engine releases and hotfixes.

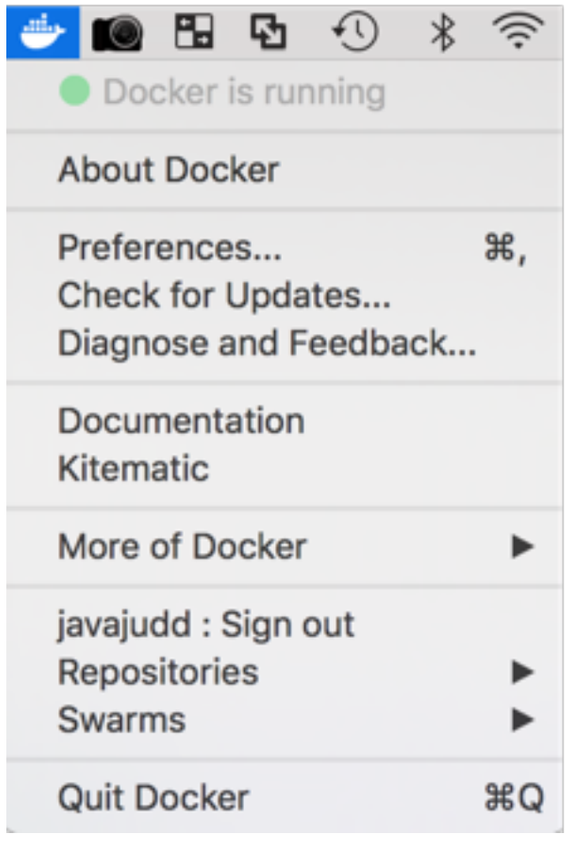
Beta channel

This installer offers cutting edge features and comes with the experimental version of Docker Engine, which is described in the Docker Experimental Features README on GitHub.

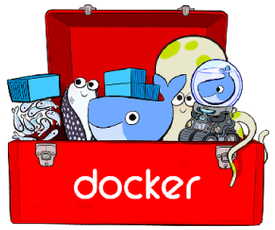
This is the best channel to use if you want to experiment with features we are working on as they become available, and can weather some instability and bugs. This channel is a

https://docs.docker.com/docker-for-mac/


xhyve
daemon
container 1
image 1
localhost


Docker is running
About Docker
Preferences... ⌘,
Check for Updates...
Diagnose and Feedback...
Documentation
Kitematic
More of Docker ▶
javajudd : Sign out
Repositories ▶
Swarms ▶
Quit Docker ⌘Q

host



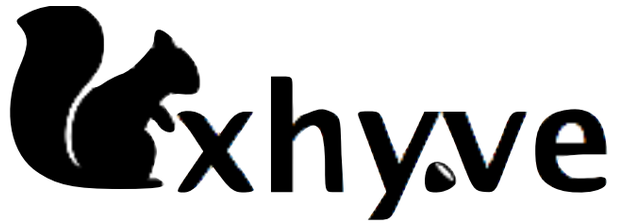
default

daemon

container 1

image 1

IP 192.168.99.100



daemon

container 1

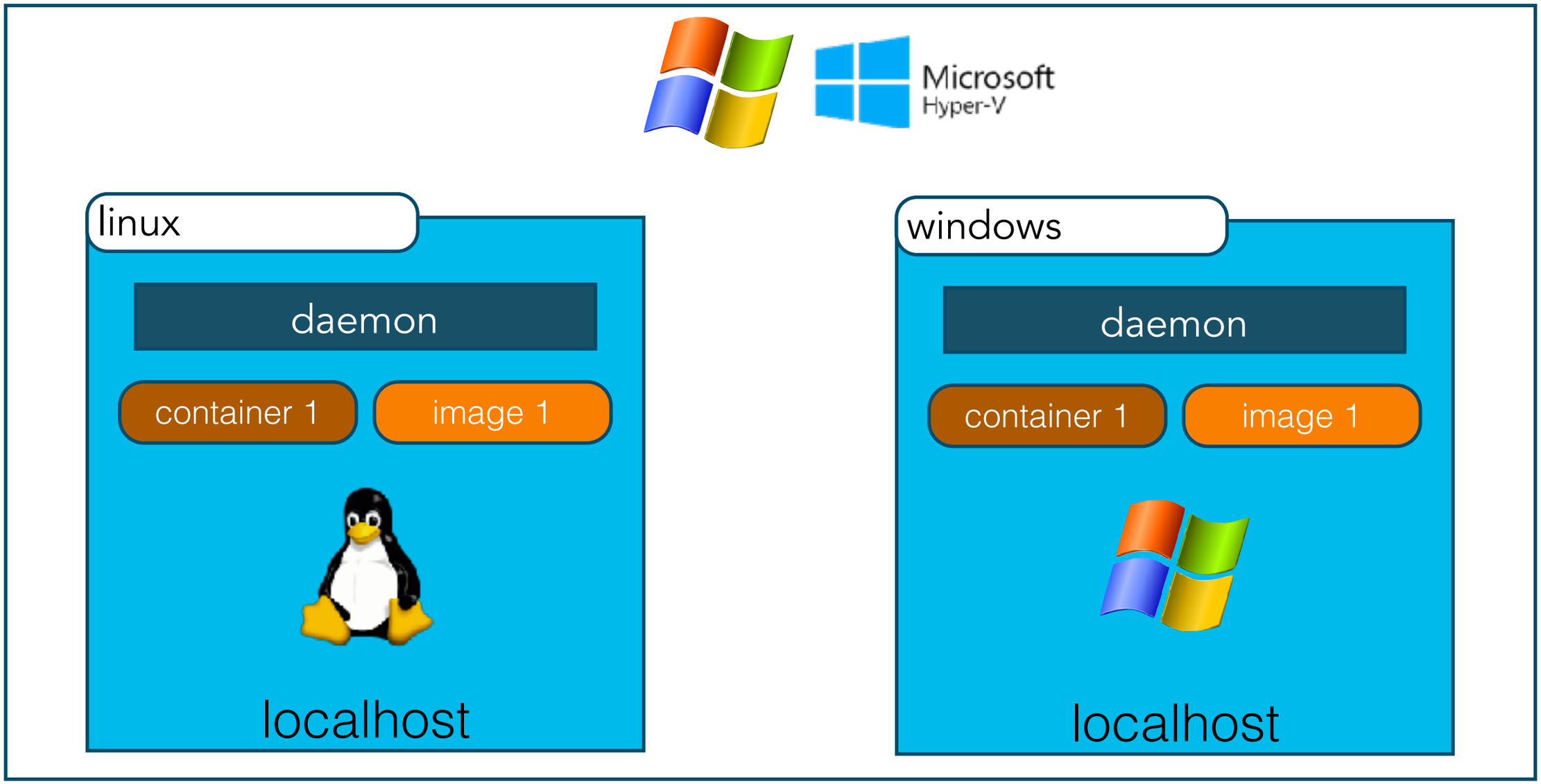
image 1

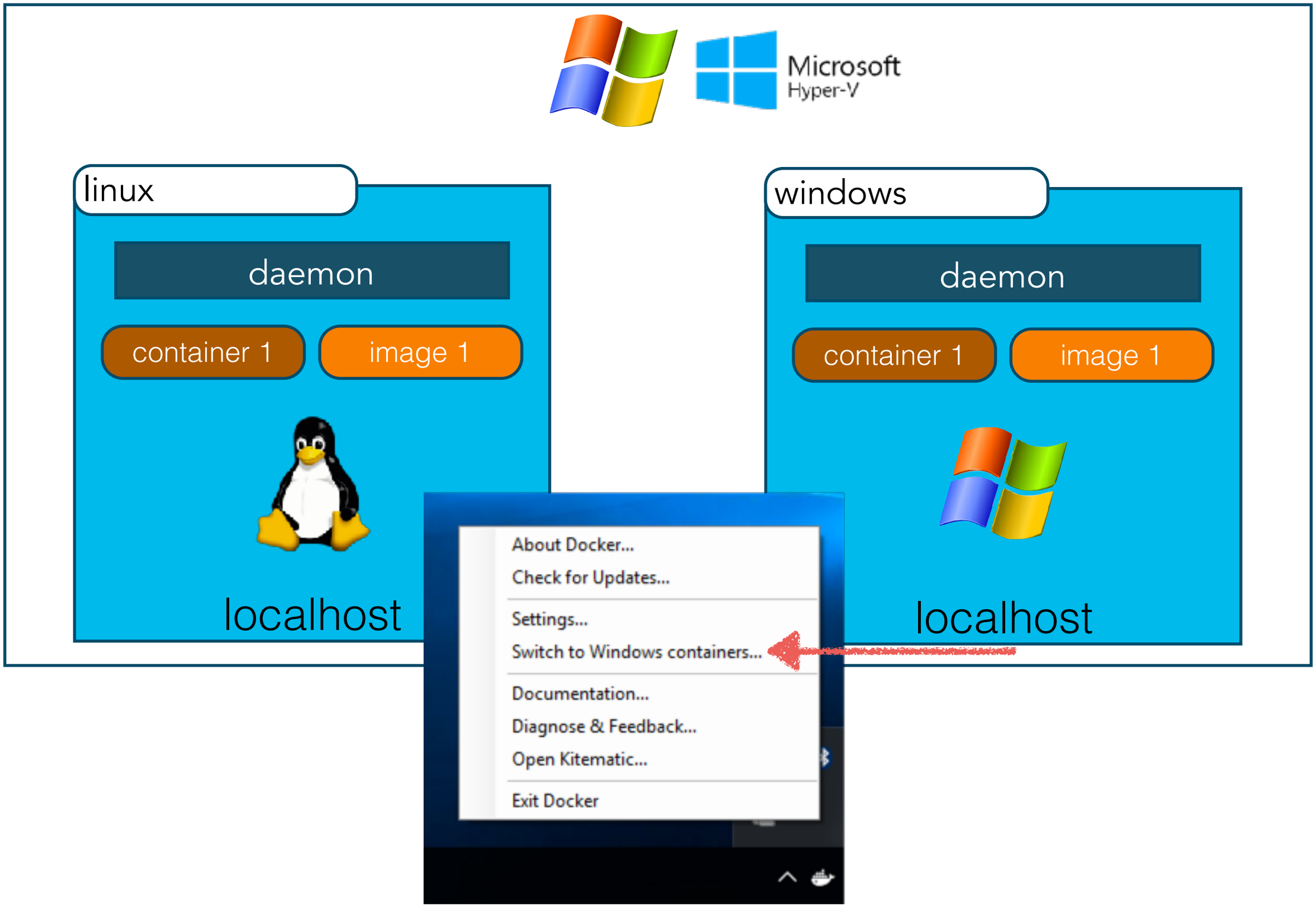
localhost

```
eval "$(docker-machine env default)"
```

```
unset ${!DOCKER_*}
```

https://docs.docker.com/docker-for-windows/

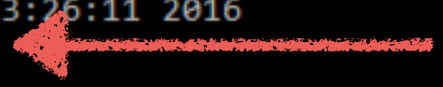

Microsoft
Hyper-V
linux
daemon
container 1
image 1
localhost
windows
daemon
container 1
image 1
localhost


Microsoft
Hyper-V
linux
daemon
container 1
image 1
localhost
windows
daemon
container 1
image 1
localhost
About Docker...
Check for Updates...
Settings...
Switch to Windows containers...
Documentation...
Diagnose & Feedback...
Open Kitematic...
Exit Docker

Recycle Bin

Command Prompt

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Christopher Judd>docker version
Client:
 Version:      1.12.3
 API version:  1.24
 Go version:   go1.6.3
 Git commit:   6b644ec
 Built:        Wed Oct 26 23:26:11 2016
 OS/Arch:      windows/amd64

Server:
 Version:      1.12.3
 API version:  1.24
 Go version:   go1.6.3
 Git commit:   6b644ec
 Built:        Wed Oct 26 23:26:11 2016
 OS/Arch:      linux/amd64

C:\Users\Christopher Judd>_
```



Windows PowerShell

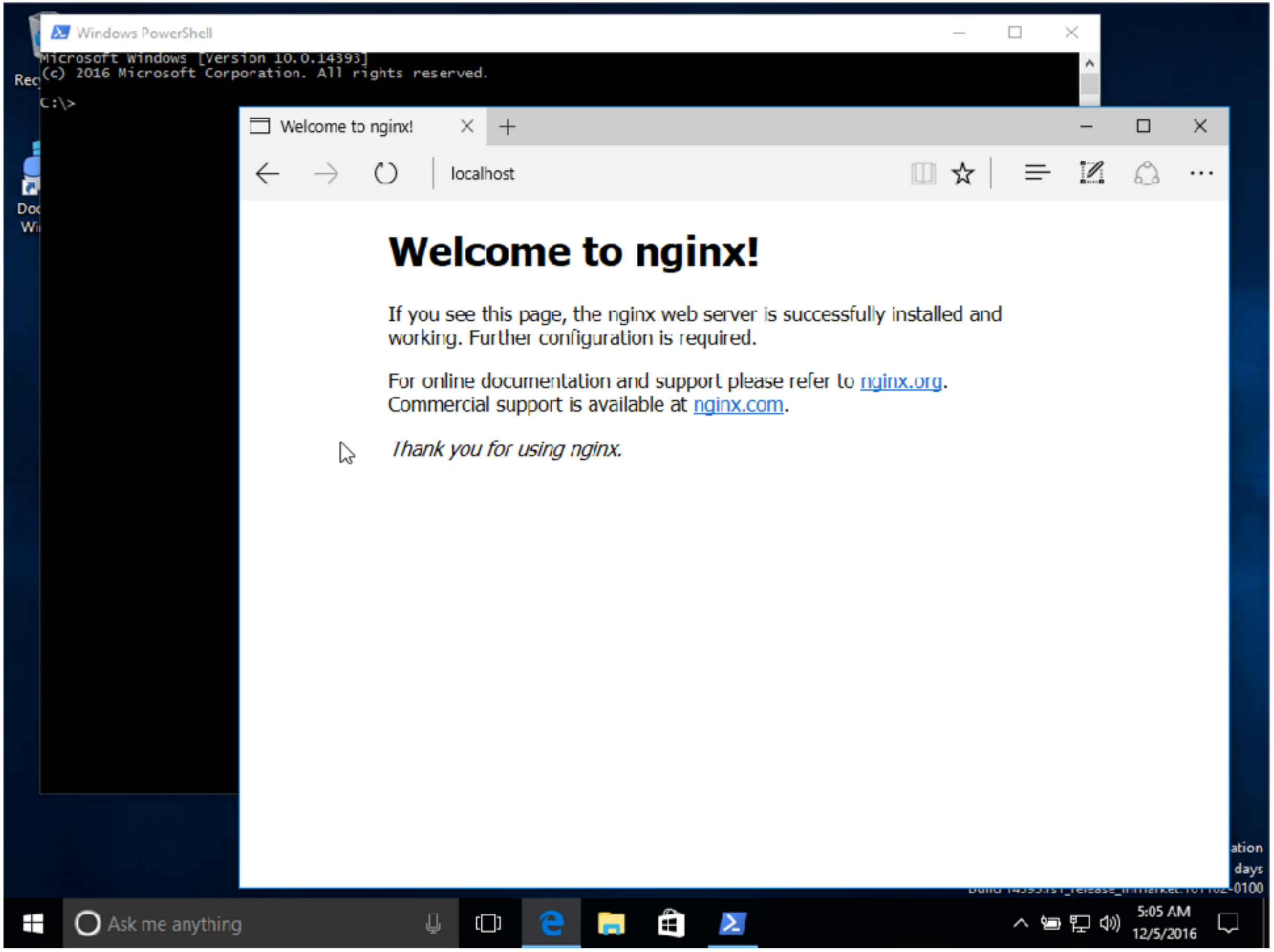
```
PS C:\Users\Christopher Judd> docker version
Client:
 Version:      1.13.0-rc2
 API version:  1.25
 Go version:   go1.7.3
 Git commit:   1f9b3ef
 Built:        Wed Nov 23 17:40:58 2016
 OS/Arch:      windows/amd64

Server:
 Version:              1.13.0-rc2
 API version:          1.25
 Minimum API version:  1.24
 Go version:           go1.7.3
 Git commit:           1f9b3ef
 Built:                Wed Nov 23 17:40:58 2016
 OS/Arch:              windows/amd64
 Experimental:         false
PS C:\Users\Christopher Judd>
```

Windows 10 Enterprise Evaluation
Windows License valid for 90 days
Build 14393.rs1_release_inmarket.161102-0100

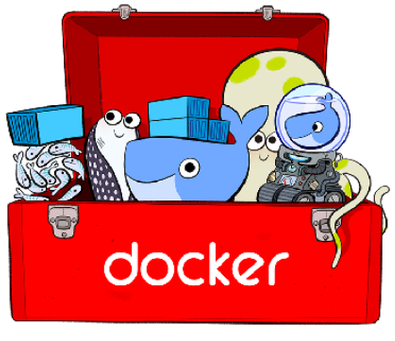
Microsoft Edge

Ask me anything

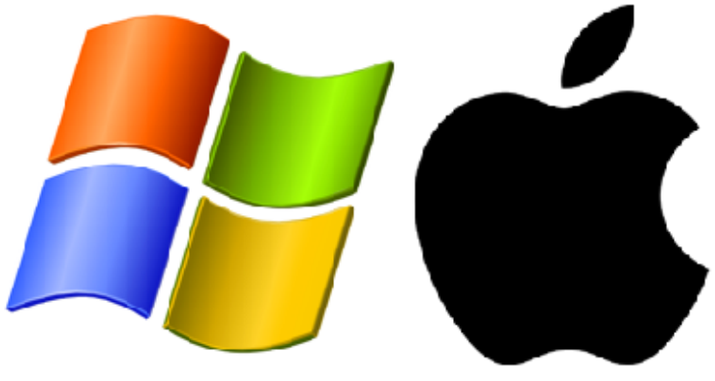
5:07 AM
12/5/2016




```
docker run -e "ASPNETCODE_URLS=http://+:80" -it --rm microsoft/dotnet:nanoserver
```



VS



- works on older OSs
- docker compose & docker machine
- swarm

- use localhost
- less overhead

Download Docker Community Edition

Developer Desktops

DOCKER CE FOR MAC

An integrated, easy-to-deploy Docker development environment on the Mac for building, assembling, and shipping applications.

Download from Docker Store Learn More

DOCKER CE FOR WINDOWS

A native Windows desktop application to easily setup a Docker development environment on a Windows PC.

Download from Docker Store Learn More

Linux Servers

DOCKER CE FOR CENTOS DISTRIBUTION

Installer for CentOS distribution environments
Also available with support and certification with Docker EE for CentOS distribution.

Download from Docker Store Learn More

DOCKER CE FOR DEBIAN

Installer for Debian environments

Download from Docker Store Learn More

DOCKER CE FOR FEDORA

Installer for Fedora environments

Looking for enterprise Linux? is available as certified infrastructure with support.

Download from Docker Store Learn More

DOCKER CE FOR UBUNTU

Installer for Ubuntu environments

Also available with support and certification with Docker EE for Ubuntu.

Download from Docker Store Learn More

IaaS Cloud

DOCKER CE FOR AWS

A native AWS application optimized and integrated to take advantage of the underlying AWS IaaS services for a modern Docker environment. Be up and running in a few clicks with no extra software to install.

Also available with integrated container management, security and support for enterprise teams with

Launch Stack from Docker Store Learn More

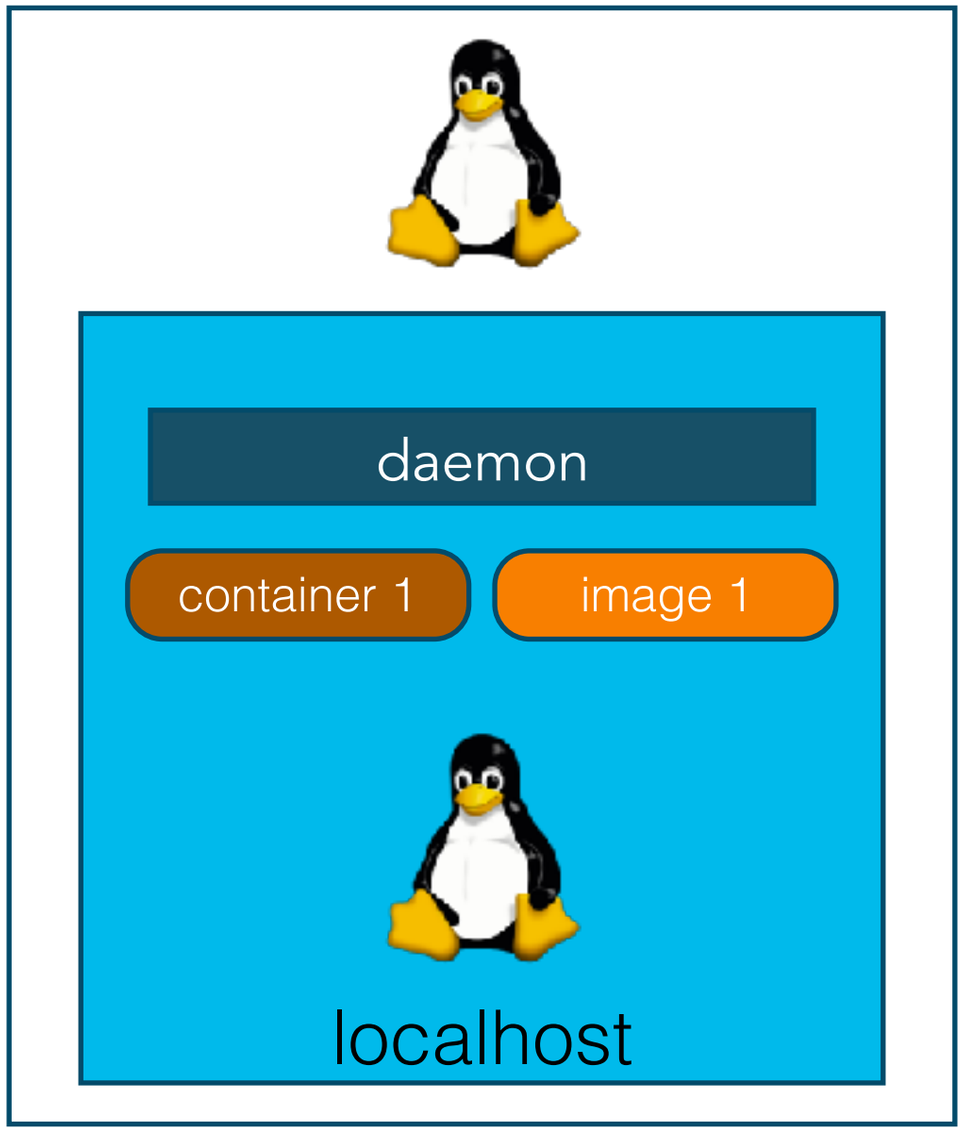
DOCKER CE FOR AZURE

Set up Docker in a few clicks with this native Azure app optimized for and integrated to the underlying Azure IaaS services. Accelerate time to productivity in building and running Docker apps in Azure.

Also available with integrated container management, security and support for enterprise teams with

Launch Template from Docker Store Learn More

https://www.docker.com/community-edition


daemon
container 1
image 1
localhost

docker version

```
Client:
 Version:  17.12.0-ce
 API version:  1.35
 Go version:   go1.9.2
 Git commit:   c97c6d6
 Built: Wed Dec 27 20:03:51 2017
 OS/Arch:  darwin/amd64

Server:
 Engine:
  Version: 17.12.0-ce
  API version: 1.35 (minimum version 1.12)
  Go version:  go1.9.2
  Git commit:  c97c6d6
  Built:   Wed Dec 27 20:12:29 2017
  OS/Arch: linux/amd64
  Experimental:true
```

```
docker-machine version
```

```
docker-machine version 0.13.0, build 9ba6da9
```

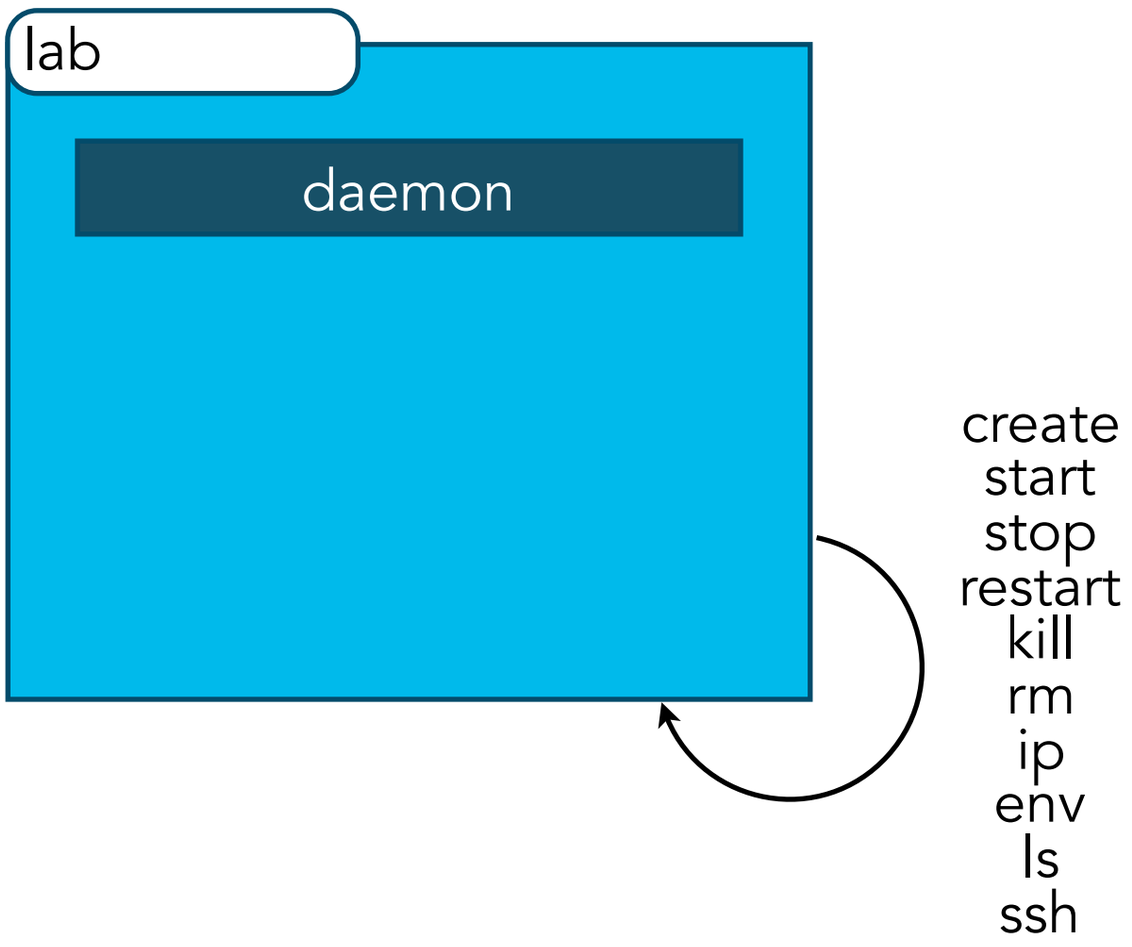
Lab 1

1. Install Docker for your specific platform

Create Machine

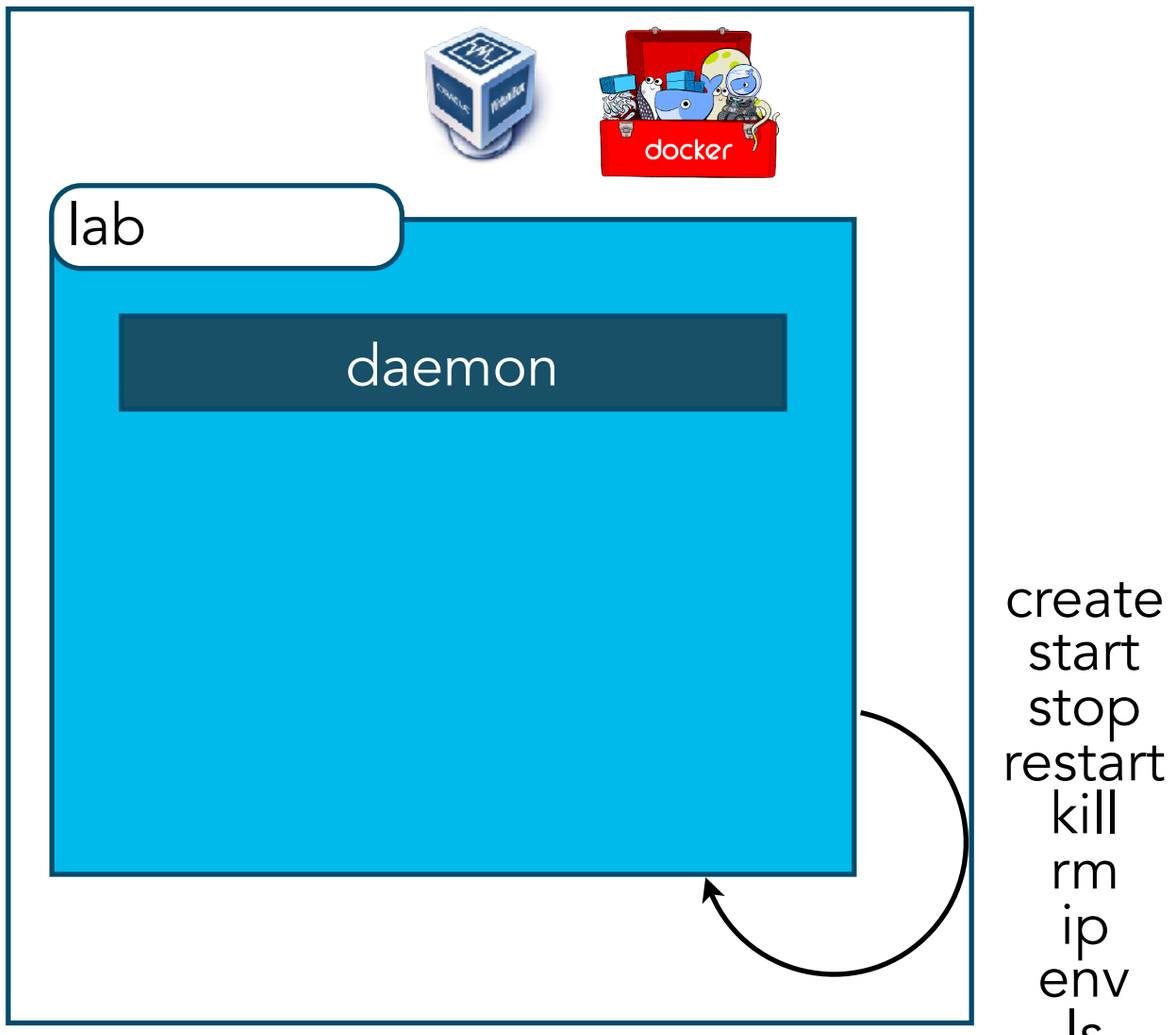
only

docker Lifecycle




```
docker-machine [OPTIONS] COMMAND [arg…]
```

docker Lifecycle




```
docker-machine [OPTIONS] COMMAND [arg…]
```

```
docker-machine create --driver=virtualbox lab
```

```
Creating VirtualBox VM...
Creating SSH key...
Starting VirtualBox VM...
Starting VM...
To see how to connect Docker to this machine, run: docker-machine env lab
```

docker-machine ls

```
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM
default             virtualbox   Stopped
dev                 virtualbox   Running   tcp://192.168.99.101:2376
client1             virtualbox   Running   tcp://192.168.99.100:2376
client2             virtualbox   Stopped
lab                 virtualbox   Running   tcp://192.168.99.102:2376
```

docker-machine env lab

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.102:2376"
export DOCKER_CERT_PATH="/Users/user/.docker/machine/machines/lab"
export DOCKER_MACHINE_NAME="lab"
# Run this command to configure your shell:
# eval "$(docker-machine env lab)"
```

eval "$(docker-machine env lab)"

docker-machine ls

NAME	ACTIVE	DRIVER	STATE	URL	SWARM
default		virtualbox	Stopped		
dev		virtualbox	Running	tcp://192.168.99.101:2376	
client1		virtualbox	Running	tcp://192.168.99.100:2376	
client2		virtualbox	Stopped		
lab	*	virtualbox	Running	tcp://192.168.99.102:2376	

docker-machine env lab

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.102:2376"
export DOCKER_CERT_PATH="/Users/user/.docker/machine/machines/lab"
export DOCKER_MACHINE_NAME="lab"
# Run this command to configure your shell:
# eval "$(docker-machine env lab)"
```


```
eval "$(docker-machine env la
```

```
docker-machine ls
```

```
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM
default             virtualbox   Stopped
dev                 virtualbox   Running   tcp://192.168.99.101:2376
client1             virtualbox   Running   tcp://192.168.99.100:2376
client2             virtualbox   Stopped
lab       *         virtualbox   Running   tcp://192.168.99.102:2376
```

```
docker-machine env --shell cmd lab
docker-machine env --shell powershell lab
docker-machine env --shell powershell lab | Invoke-Expression
```

Oracle VM VirtualBox Manager

New Settings Discard Start

Details Snapshots

- **Hadoop Playground** — Saved
- **musicbrainz** — Powered Off
- **Hadoop Tutorial 2015** — Powered Off
- **boot2docker-vm** — Powered Off
- **dev** — Aborted
- [redacted] — Running
- **default** — Powered Off
- [redacted] — Powered Off
- **lab** — Running

General

Name:	Hadoop Playground
Operating System:	Ubuntu (64-bit)

System

Base Memory:	2048 MB
Boot Order:	Floppy, Optical, Hard Disk
Acceleration:	VT-x/AMD-V, Nested Paging

Preview

Display

Video Memory:	12 MB
Remote Desktop Server:	Disabled
Video Capture:	Disabled

Storage

Controller: IDE	
IDE Secondary Master:	[Optical Drive] Empty
Controller: SATA	
SATA Port 0:	Hadoop Playground.vdi (Normal, 8.00 GB)

Audio

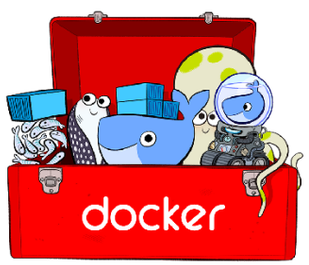
Host Driver:	CoreAudio
Controller:	ICH AC97

Network

Adapter 1:	Intel PRO/1000 MT Desktop (NAT)
Adapter 2:	Intel PRO/1000 MT Desktop (Host-only Adapter, 'vboxnet0')


```
alias dm='docker-machine'
alias dmlab='docker-machine env lab && eval "$(docker-machine env lab)"'
```

Lab 2

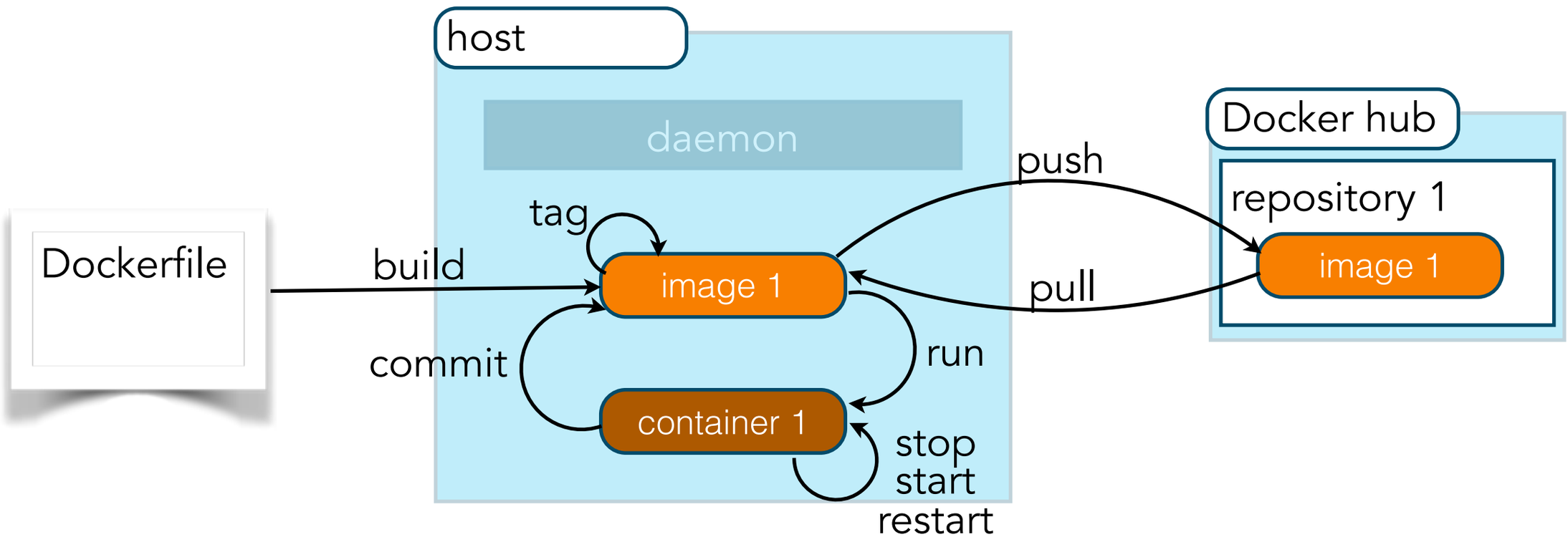


only

1. Create a new Docker Machine called lab

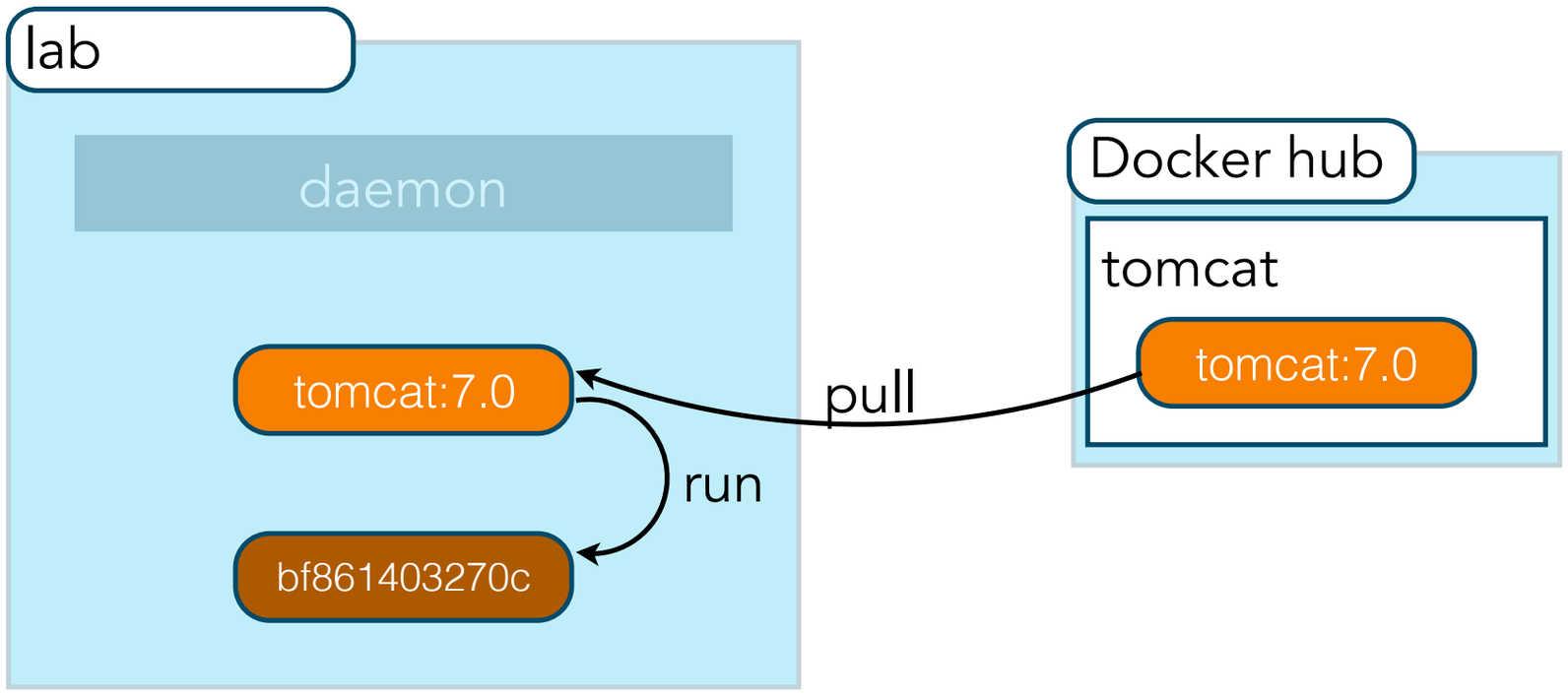
Docker Lifecycle

docker Lifecycle




```
docker [OPTIONS] COMMAND [arg…]
```

Finding & Running Containers


lab
daemon
tomcat:7.0
run
bf861403270c
pull
Docker hub
tomcat
tomcat:7.0

Explore Help

tomcat

Sign up Log In

Repositories (1181)

All

tomcat official	297 STARS	2.0 M PULLS	DETAILS
cloudesire/tomcat public | automated build	3 STARS	2.1 K PULLS	DETAILS
dordoka/tomcat public | automated build	8 STARS	6.1 K PULLS	DETAILS
inspectit/tomcat public | automated build	0 STARS	236 PULLS	DETAILS
cirit/tomcat public | automated build	1 STARS	341 PULLS	DETAILS
andreptb/tomcat public | automated build	1 STARS	244 PULLS	DETAILS

https://hub.docker.com/

Explore Help tomcat Sign up Log I

OFFICIAL REPOSITORY

tomcat ☆

Last pushed: 4 days ago

Repo Info Tags

Short Description

Apache Tomcat is an open source implementation of the Java Servlet and JavaServer Pages technologies

Docker Pull Command

```
docker pull tomcat
```

Full Description

Supported tags and respective `Dockerfile` links

- `6.0.44-jre7`, `6.0-jre7`, `6-jre7`, `6.0.44`, `6.0`, `6` (*6-jre7/Dockerfile*)
- `6.0.44-jre8`, `6.0-jre8`, `6-jre8` (*6-jre8/Dockerfile*)
- `7.0.64-jre7`, `7.0-jre7`, `7-jre7`, `7.0.64`, `7.0`, `7` (*7-jre7/Dockerfile*)
- `7.0.64-jre8`, `7.0-jre8`, `7-jre8` (*7-jre8/Dockerfile*)
- `8.0.28-jre7`, `8.0-jre7`, `8-jre7`, `jre7`, `8.0.28`, `8.0`, `8`, `latest` (*8-jre7/Dockerfile*)
- `8.0.28-jre8`, `8.0-jre8`, `8-jre8`, `jre8` (*8-jre8/Dockerfile*)

For more information about this image and its history, please see the relevant manifest file

How to use this image.

Run the default Tomcat server (`CMD ["catalina.sh", "run"]`):

```
$ docker run -it --rm tomcat:8.0
```

You can test it by visiting `http://container-ip:8080` in a browser or, if you need access outside the host, on port 8888:

```
$ docker run -it --rm -p 8888:8080 tomcat:8.0
```

You can then go to `http://localhost:8888` or `http://host-ip:8888` in a browser.

The default Tomcat environment in the image for versions 7 and 8 is:

```
CATALINA_BASE:   /usr/local/tomcat
CATALINA_HOME:   /usr/local/tomcat
CATALINA_TMPDIR: /usr/local/tomcat/temp
JRE_HOME:        /usr
CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bi
```

The default Tomcat environment in the image for version 6 is:

```
CATALINA_BASE:   /usr/local/tomcat
CATALINA_HOME:   /usr/local/tomcat
CATALINA_TMPDIR: /usr/local/tomcat/temp
JRE_HOME:        /usr
CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar
```

The configuration files are available in `/usr/local/tomcat/conf/` . By default, no user is included in the "manager-gui" role required to operate the "/manager/html" web application. If you wish to use this app, you must define such a user in `tomcat-users.xml` .

License

View license information for the software contained in this image.

This image is officially supported on Docker version 1.8.3.

Support for older versions (down to 1.6) is provided on a best-effort basis.

Please see the Docker installation documentation for details on how to upgrade your Docker daemon.

User Feedback

Documentation

Documentation for this image is stored in the `tomcat/` directory of the `docker-library/docs` GitHub repo. Be sure to familiarize yourself with the repository's `README.md` file before attempting a pull request.

Issues

If you have any problems with or questions about this image, please contact us through a GitHub issue.

You can also reach many of the official image maintainers via the `#docker-library` IRC channel on Freenode.

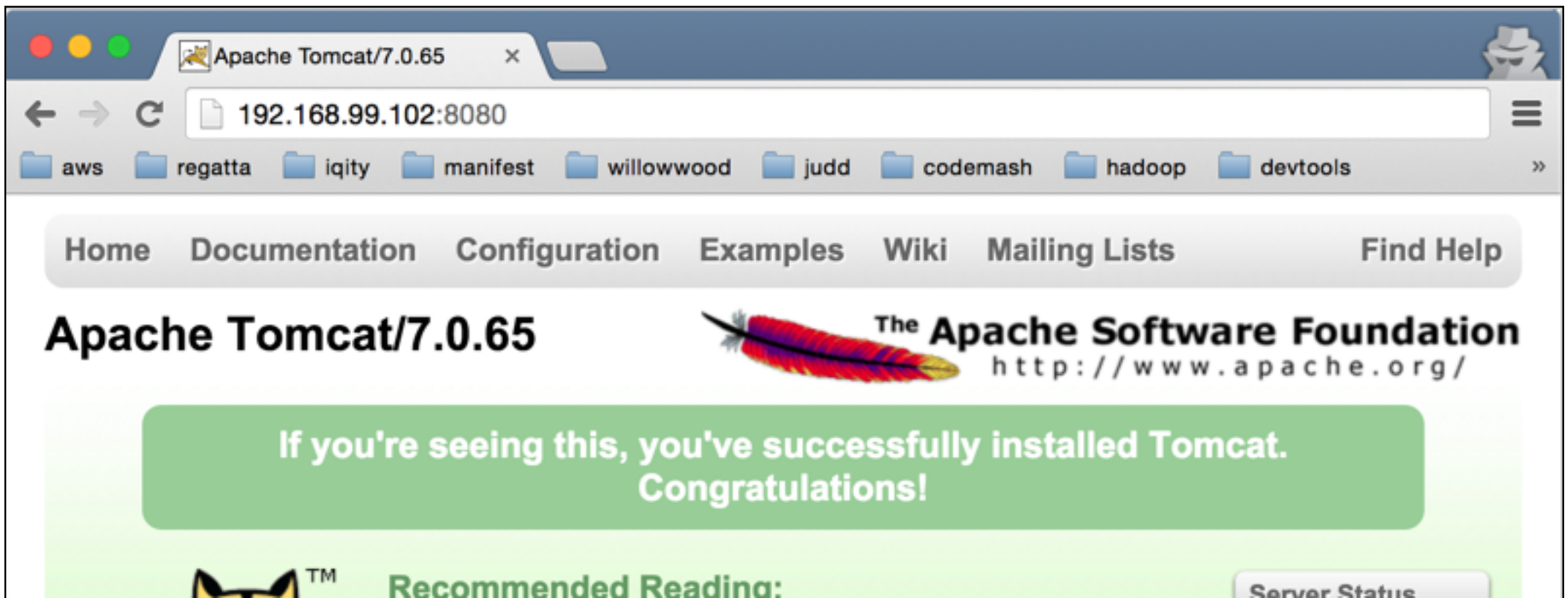
Contributing

You are invited to contribute new features, fixes, or updates, large or small; we are always thrilled to receive pull requests, and do our best to process them as fast as we can.

Before you start to code, we recommend discussing your plans through a GitHub issue, especially for more ambitious contributions. This gives other contributors a chance to point you in the right direction, give you feedback on your design, and help you find out if someone else is working on the same thing.

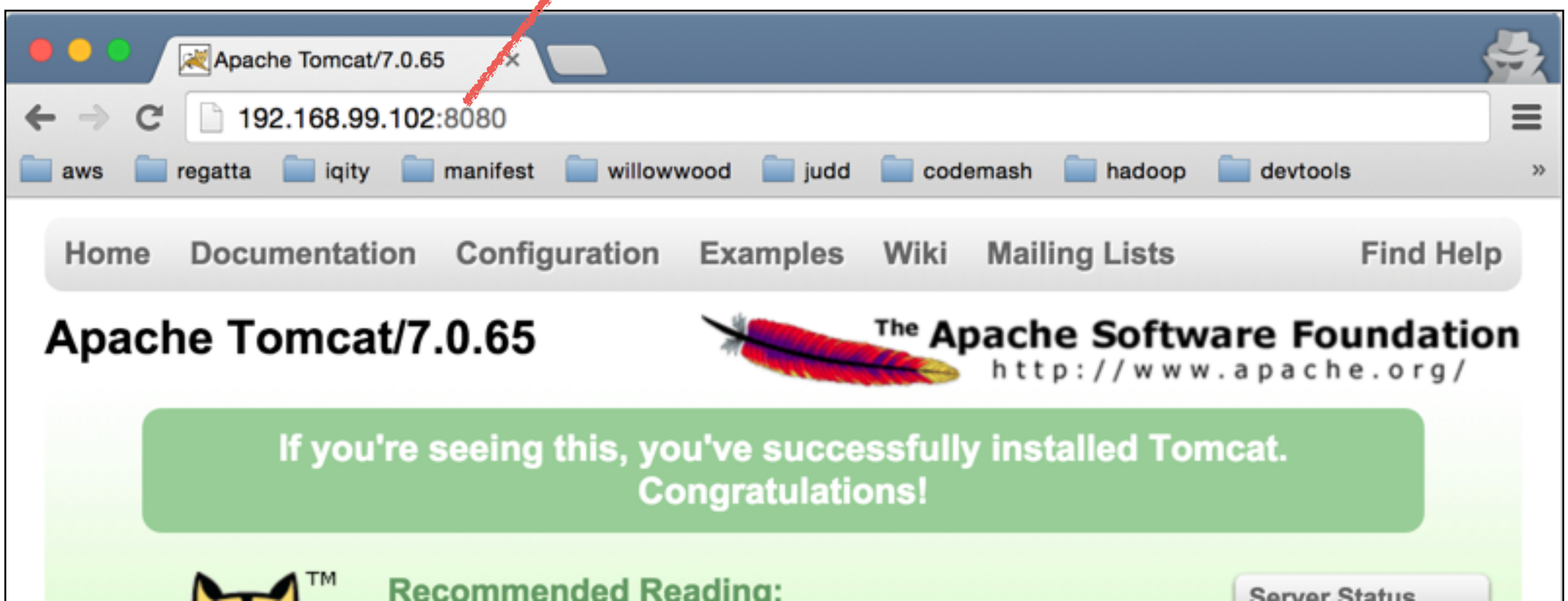
docker run -it --rm -p 8080:8080 tomcat:7.0-jre7

```
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar
18-Oct-2015 23:34:47.365 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version:        Apache Tomcat/8.0.28
18-Oct-2015 23:34:47.366 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built:          Oct 7 2015 18:25:21 UTC
18-Oct-2015 23:34:47.367 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server number:        8.0.28.0
18-Oct-2015 23:34:47.367 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name:              Linux
18-Oct-2015 23:34:47.368 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version:           4.0.9-boot2docker
18-Oct-2015 23:34:47.368 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture:          amd64
18-Oct-2015 23:34:47.369 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home:            /usr/lib/jvm/java-7-openjdk-amd64/jre
18-Oct-2015 23:34:47.369 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version:          1.7.0_79-b14
18-Oct-2015 23:34:47.370 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor:           Oracle Corporation
18-Oct-2015 23:34:47.370 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE:        /usr/local/tomcat
18-Oct-2015 23:34:47.371 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME:        /usr/local/tomcat
18-Oct-2015 23:34:47.371 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.config.file=/usr/local/tomcat/
conf/logging.properties
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.endorsed.dirs=/usr/local/tomcat/endorsed
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.base=/usr/local/tomcat
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.home=/usr/local/tomcat
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.io.tmpdir=/usr/local/tomcat/temp
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent The APR based Apache Tomcat Native library which allows optimal
performance in production environments was not found on the java.library.path: /usr/java/packages/lib/amd64:/usr/lib/x86_64-linux-gnu/jni:/lib/x86_64-linux-gnu:/usr/
lib/x86_64-linux-gnu:/usr/lib/jni:/lib:/usr/lib
18-Oct-2015 23:34:47.519 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-nio-8080"]
18-Oct-2015 23:34:47.546 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared selector for servlet write/read
18-Oct-2015 23:34:47.559 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["ajp-nio-8009"]
18-Oct-2015 23:34:47.561 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared selector for servlet write/read
18-Oct-2015 23:34:47.562 INFO [main] org.apache.catalina.startup.Catalina.load Initialization processed in 812 ms
18-Oct-2015 23:34:47.607 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service Catalina
18-Oct-2015 23:34:47.618 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet Engine: Apache Tomcat/8.0.28
18-Oct-2015 23:34:47.647 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /usr/local/tomcat/
webapps/ROOT
```



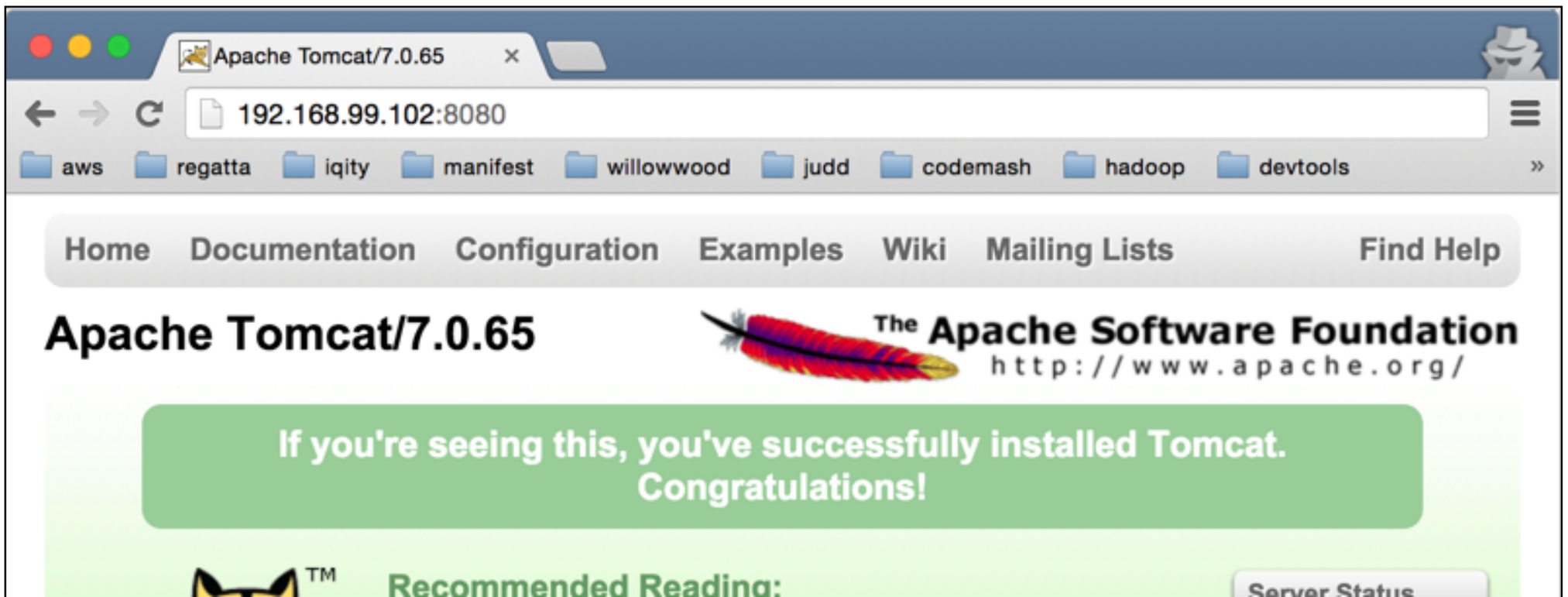
docker run -it --rm -p 8080:8080 tomcat:7.0-jre7

```
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar
18-Oct-2015 23:34:47.365 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version:        Apache Tomcat/8.0.28
18-Oct-2015 23:34:47.366 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built:          Oct 7 2015 18:25:21 UTC
18-Oct-2015 23:34:47.367 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server number:        8.0.28.0
18-Oct-2015 23:34:47.367 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name:               Linux
18-Oct-2015 23:34:47.368 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version:            4.0.9-boot2docker
18-Oct-2015 23:34:47.368 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture:          amd64
18-Oct-2015 23:34:47.369 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home:             /usr/lib/jvm/java-7-openjdk-amd64/jre
18-Oct-2015 23:34:47.369 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version:           1.7.0_79-b14
18-Oct-2015 23:34:47.370 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor:            Oracle Corporation
18-Oct-2015 23:34:47.370 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE:         /usr/local/tomcat
18-Oct-2015 23:34:47.371 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME:         /usr/local/tomcat
18-Oct-2015 23:34:47.371 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.config.file=/usr/local/tomcat/
conf/logging.properties
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.endorsed.dirs=/usr/local/tomcat/endorsed
18-Oct-2015 23:34:47.372 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.base=/usr/local/tomcat
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.home=/usr/local/tomcat
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.io.tmpdir=/usr/local/tomcat/temp
18-Oct-2015 23:34:47.373 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent The APR based Apache Tomcat Native library which allows optimal
performance in production environments was not found on the java.library.path: /usr/java/packages/lib/amd64:/usr/lib/x86_64-linux-gnu/jni:/lib/x86_64-linux-gnu:/usr/
lib/x86_64-linux-gnu:/usr/lib/jni:/lib:/usr/lib
18-Oct-2015 23:34:47.519 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-nio-8080"]
18-Oct-2015 23:34:47.546 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared selector for servlet write/read
18-Oct-2015 23:34:47.559 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["ajp-nio-8009"]
18-Oct-2015 23:34:47.561 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared selector for servlet write/read
18-Oct-2015 23:34:47.562 INFO [main] org.apache.catalina.startup.Catalina.load Initialization processed in 812 ms
18-Oct-2015 23:34:47.607 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service Catalina
18-Oct-2015 23:34:47.618 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet Engine: Apache Tomcat/8.0.28
18-Oct-2015 23:34:47.647 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /usr/local/tomcat/
webapps/ROOT
```



docker-machine ip lab

192.168.99.102

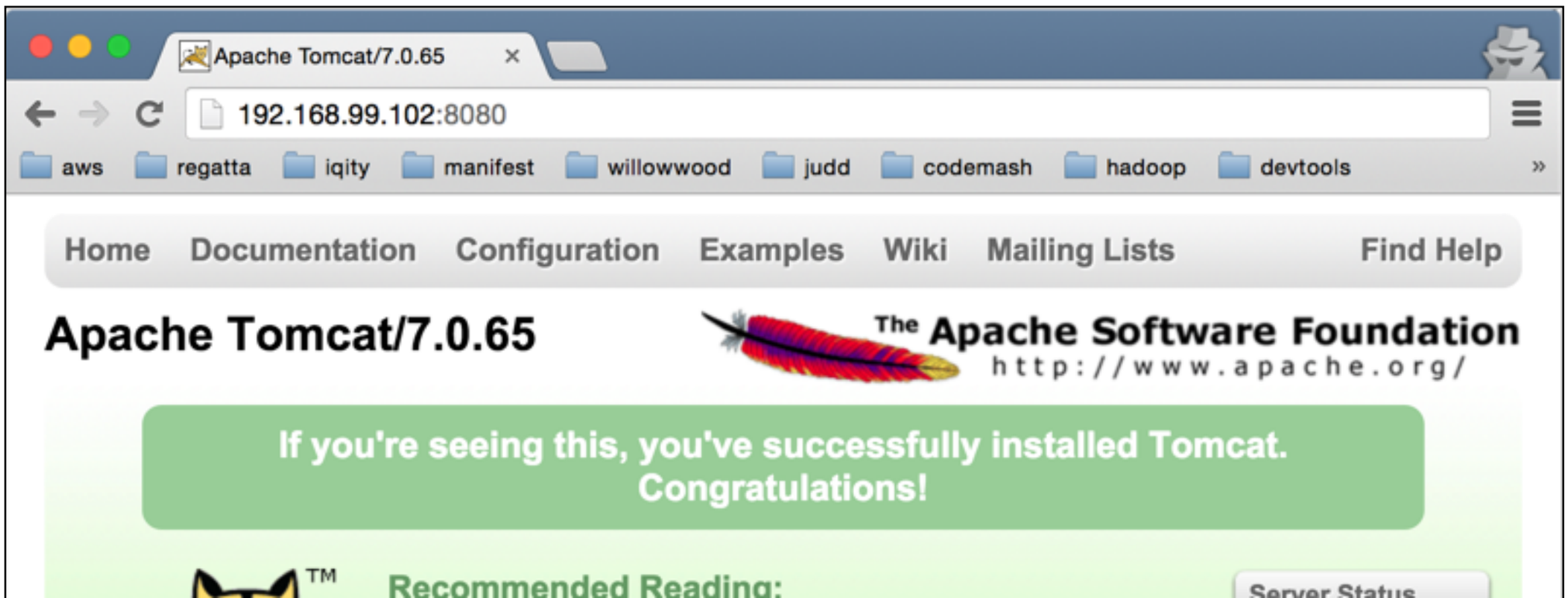


docker-machine ip lab

```
192.168.99.102
```

docker-machine ls

```
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM
default            virtualbox   Stopped
dev                virtualbox   Running   tcp://192.168.99.101:2376
client1            virtualbox   Running   tcp://192.168.99.100:2376
client2            virtualbox   Stopped
lab                virtualbox   Running   tcp://192.168.99.102:2376
```

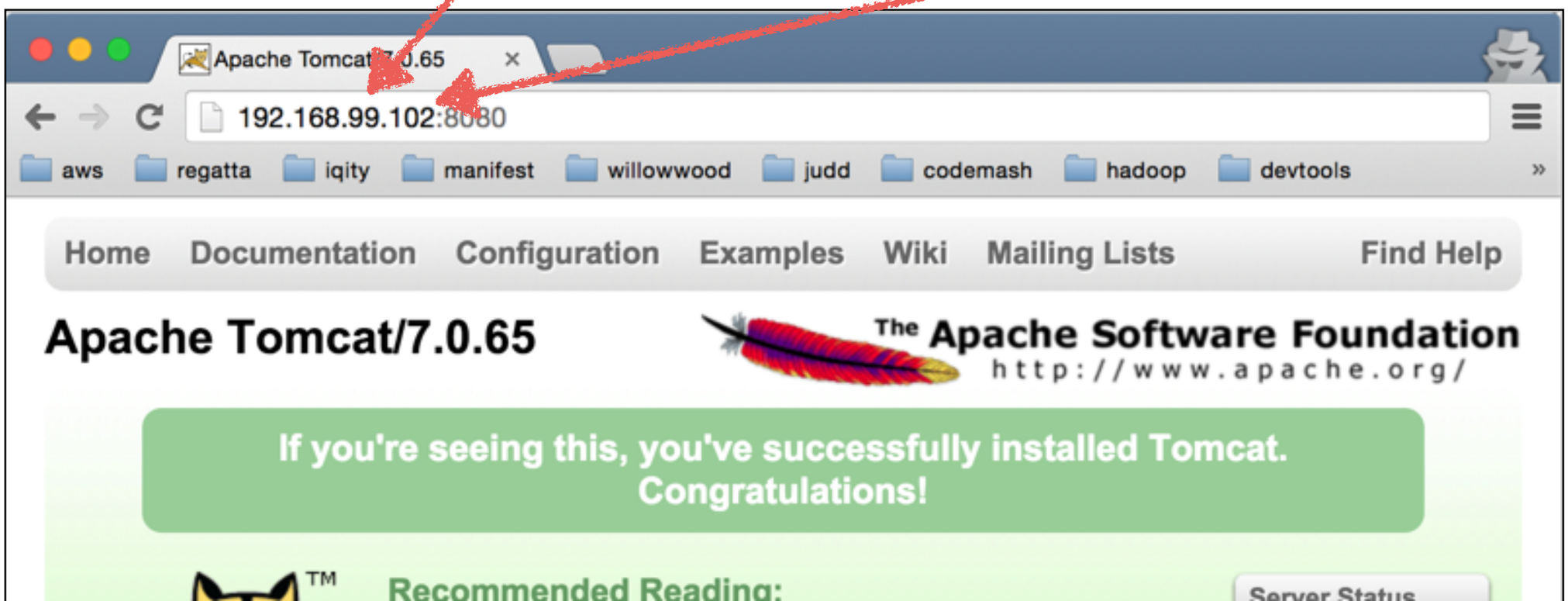


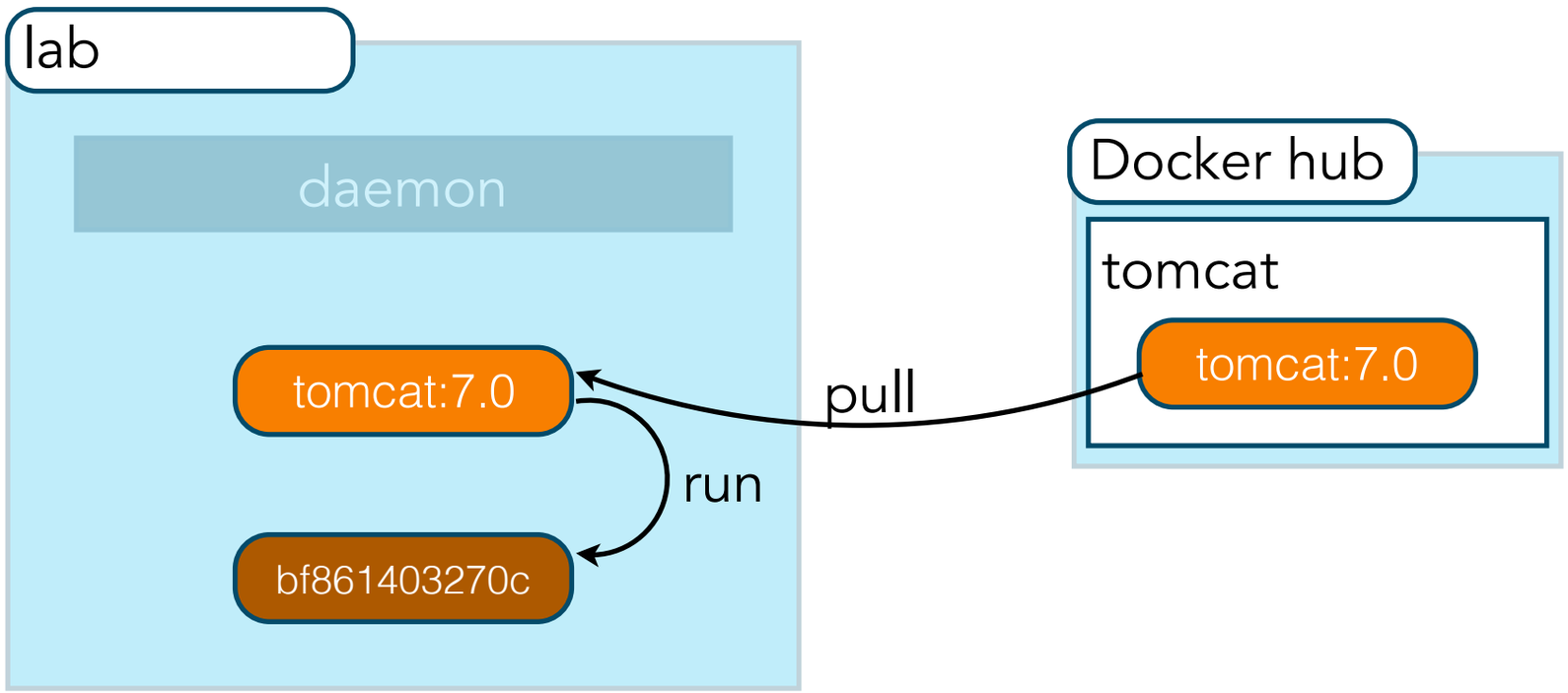
```
docker-machine ip lab
```

```
192.168.99.102
```

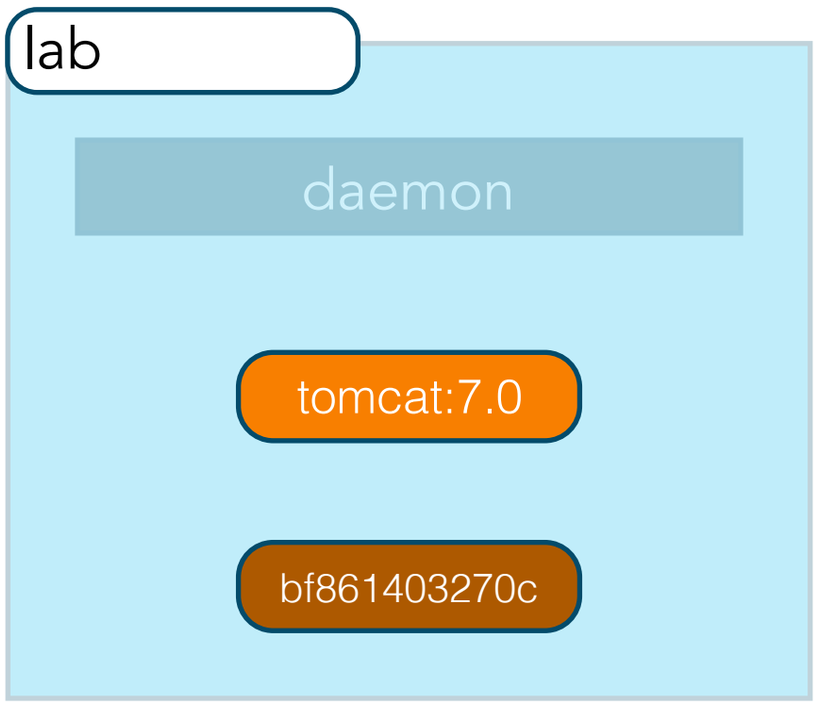
```
docker-machine ls
```

```
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM
default             virtualbox   Stopped
dev                 virtualbox   Running   tcp://192.168.99.101:2376
client1             virtualbox   Running   tcp://192.168.99.100:2376
client2             virtualbox   Stopped
lab                 virtualbox   Running   tcp://192.168.99.102:2376
```



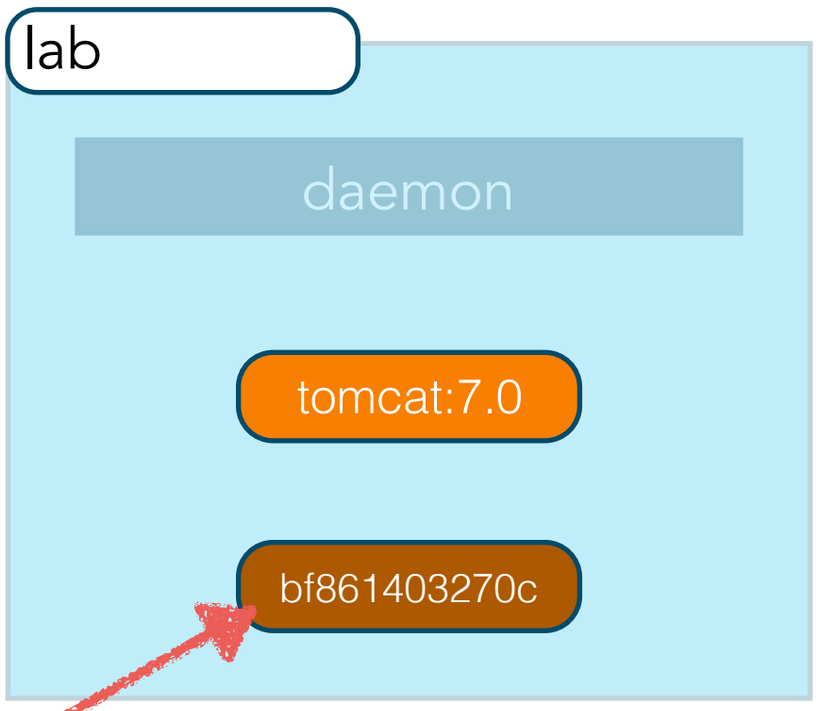



```
docker run -it --rm -p 8080:8080 tomcat:7.0-jre7
```



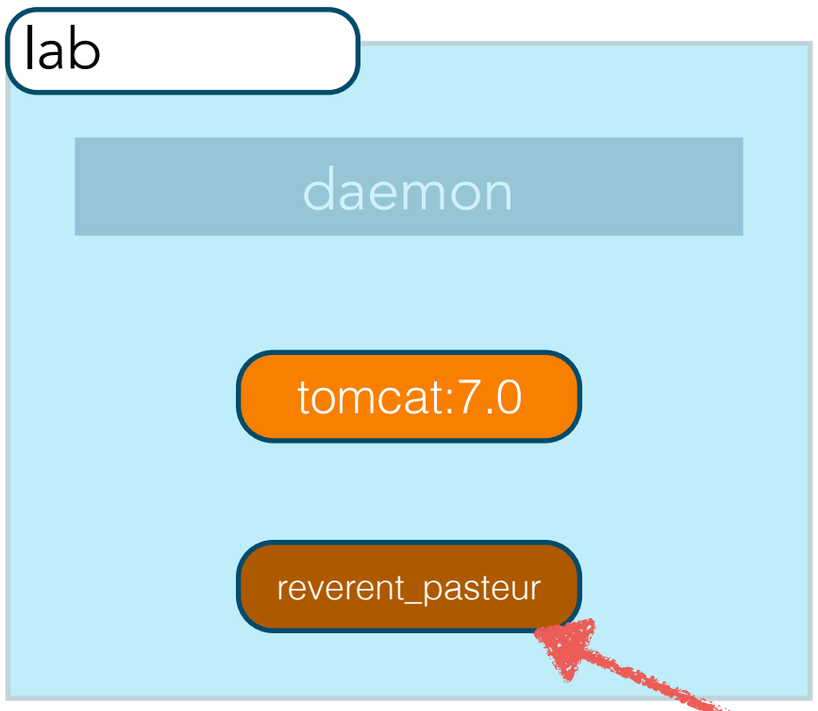

```
docker ps
```

```
CONTAINER ID  IMAGE              COMMAND             STATUS        PORTS                    NAMES
bf861403270c  tomcat:7.0-jre7  "catalina.sh run"  Up 2 seconds  0.0.0.0:8080->8080/tcp reverent_pasteur
```



docker ps

```
CONTAINER ID  IMAGE              COMMAND             STATUS         PORTS                    NAMES
bf861403270c  tomcat:7.0-jre7  "catalina.sh run"  Up 2 seconds  0.0.0.0:8080->8080/tcp  reverent_pasteur
```




```
docker ps

CONTAINER ID  IMAGE              COMMAND             STATUS         PORTS                    NAMES
bf861403270c  tomcat:7.0-jre7  "catalina.sh run"  Up 2 seconds  0.0.0.0:8080->8080/tcp reverent_pasteur
```

Lab 3

1. Run Tomcat 7.0-jre7 container
2. Validate you can see the Tomcat home page
3. Determine the name & id of your container
4. Shut the container down

Hint: you might want to use two command prompts to do this

Creating Images

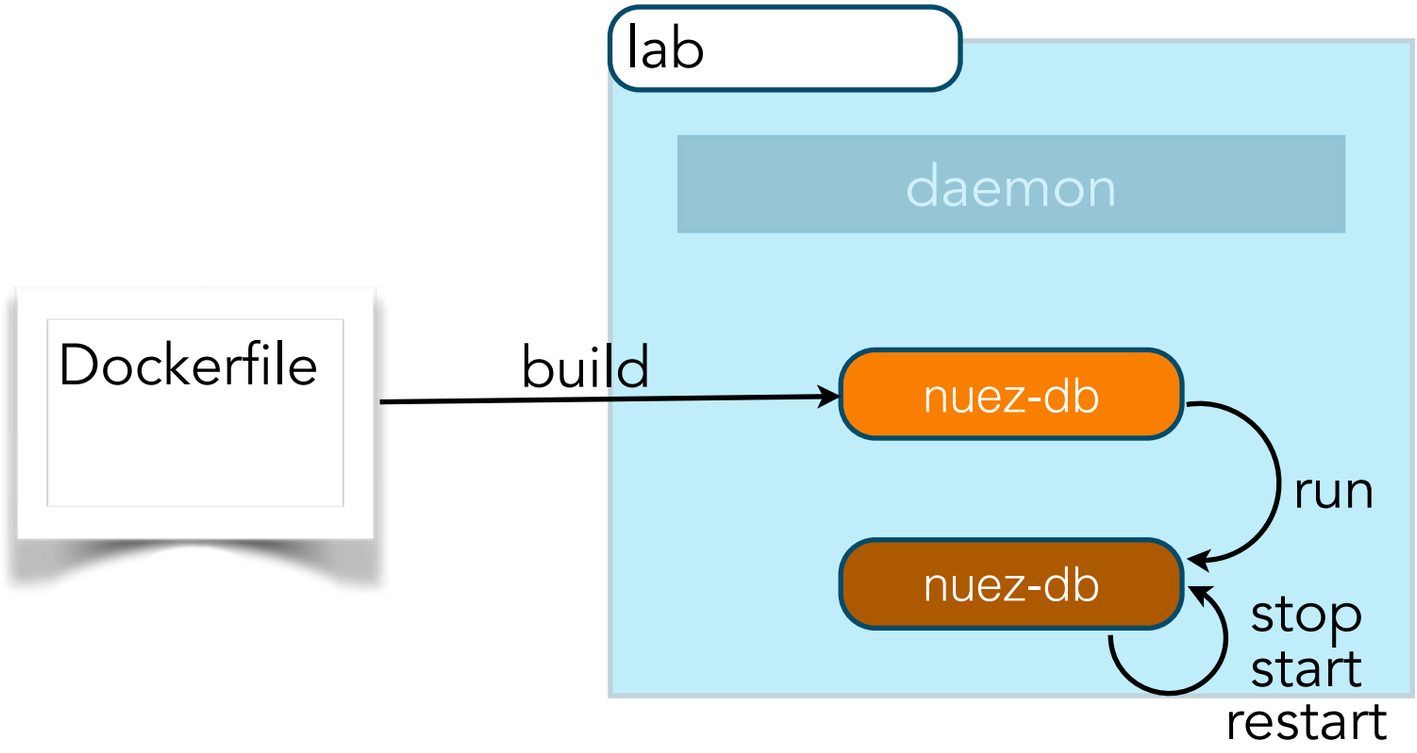
- Run container and manually install software
- Dockerfile

- ~~Run container and manually install software~~
- Dockerfile

Dockerfile

```
FROM mysql:5.5.45

RUN echo America/New_York | tee /etc/timezone && dpkg-reconfigure --frontend noninteractive tzdata
RUN echo "lower_case_table_names = 1" >> /etc/mysql/my.cnf

ENTRYPOINT ["/entrypoint.sh"]
CMD ["mysqld"]
```

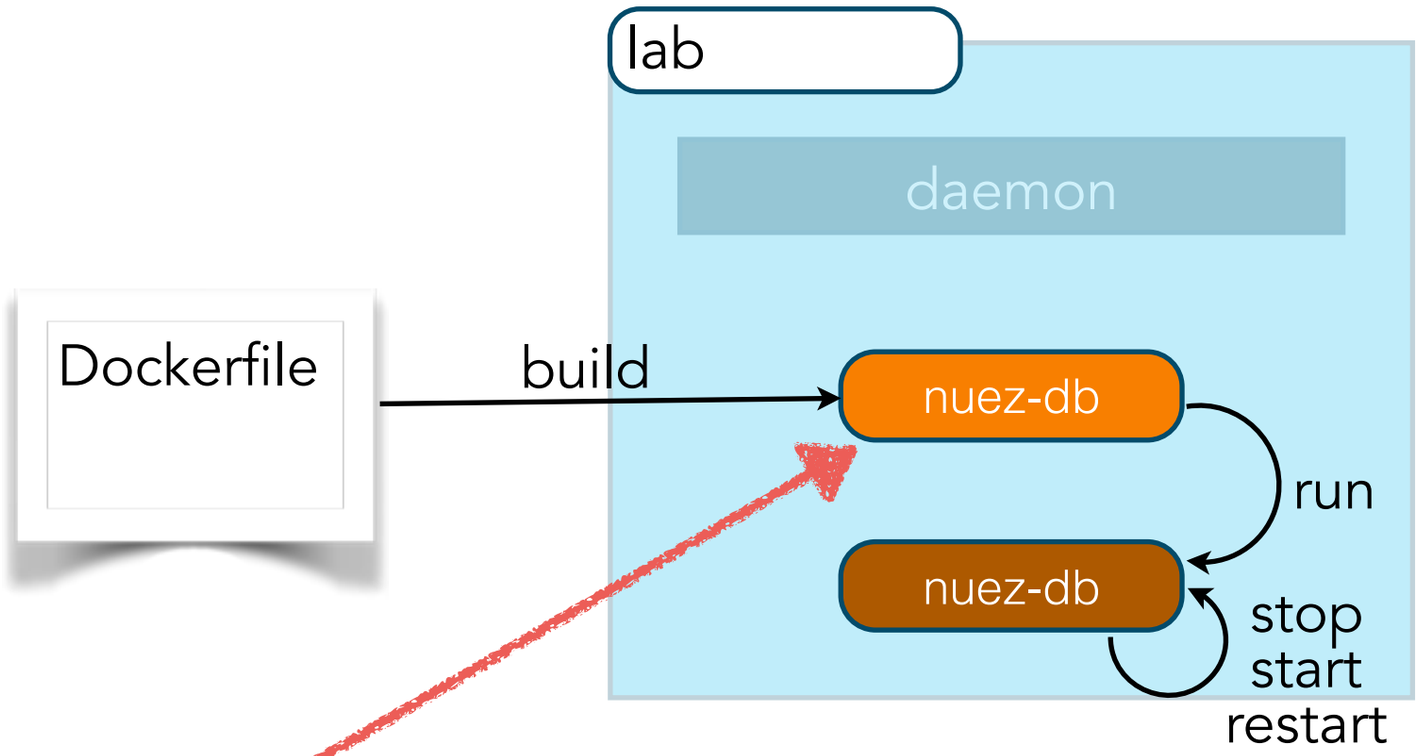
Instruction	Description
FROM	This is must be the first instruction in the Dockerfile and identifies the image to
MAINTAINER	Provides visibility and credit to the author of the image.
RUN	Executes a Linux command for configuring and installing.
ENTRYPOINT	The final script or application used bootstrap the container making it an
CMD	Provide default arguments to the ENTRYPOINT using a JSON array format.
LABEL	Name/value metadata about the image.
ENV	Sets environment variables.
COPY	Copies files into the container.
ADD	Alternative to copy.
WORKDIR	Sets working directory for RUN, CMD, ENTRYPOINT, COPY and/or ADD
EXPOSE	Ports the container will listen on.
VOLUME	Creates a mount point.
USER	User to run RUN, CMD and/or ENTRYPOIN instructions..




```
docker build -t nuez-db .
```

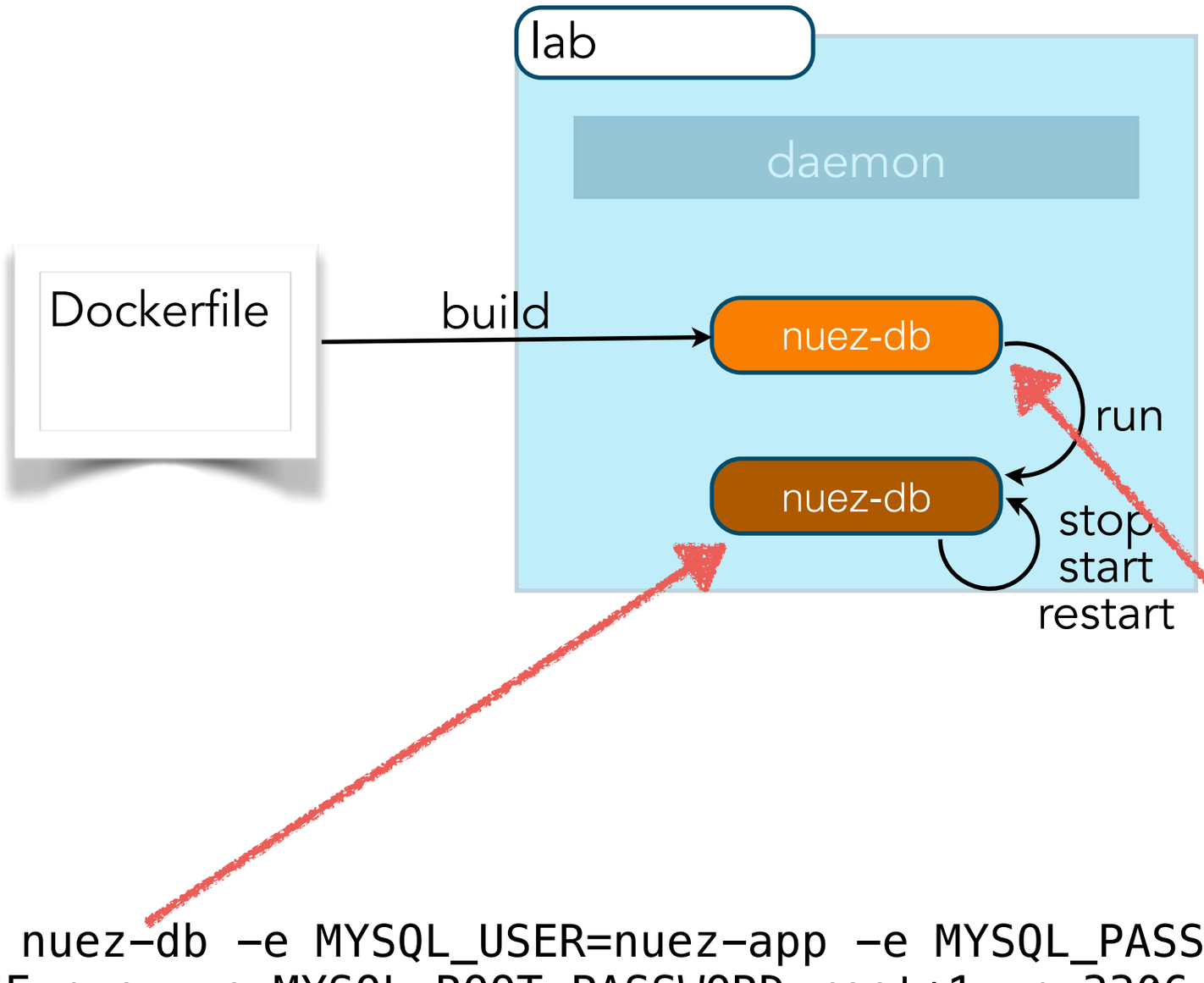
docker build -t nuez-db .

```
Sending build context to Docker daemon 35.33 kB
Step 0 : FROM mysql:5.5.45
5.5.45: Pulling from library/mysql
ba249489d0b6: Pull complete
19de96c112fc: Pull complete
2e32b26a94ed: Pull complete
637386aea7a0: Pull complete
f40aa7fe5d68: Pull complete
f9cc53679b1f: Pull complete
6b4bdd0d9a0c: Pull complete
066f73588b2c: Pull complete
8c7b0f689d1a: Pull complete
ea70677dc485: Pull complete
9d511b9dbf89: Pull complete
0896de2c270f: Pull complete
a29e02ef2642: Pull complete
bb322ed102e5: Pull complete
1c764f968123: Pull complete
2b884c0bb51d: Pull complete
0da0b10c6fd8: Pull complete
Digest: sha256:72a09a61824bdaf652e701fcbf0ee12f5b132d8fdeaf1629ce42960375de03cb
Status: Downloaded newer image for mysql:5.5.45
 ---> 0da0b10c6fd8
Step 1 : COPY docker-entrypoint-initdb.d /docker-entrypoint-initdb.d/
 ---> 46223bd058b1
Removing intermediate container bcbb6f31fed3
Step 2 : RUN echo America/New_York | tee /etc/timezone && dpkg-reconfigure --frontend noninteractive tzdata
 ---> Running in cb82914c1be9
America/New_York

Current default time zone: 'America/New_York'
Local time is now:      Sun Oct 18 20:53:55 EDT 2015.
Universal Time is now:  Mon Oct 19 00:53:55 UTC 2015.

 ---> 928355f67c39
Removing intermediate container cb82914c1be9
Step 3 : RUN echo "lower_case_table_names = 1" >> /etc/mysql/my.cnf
 ---> Running in 6ccd82a7286c
 ---> 6d23b9680c54
Removing intermediate container 6ccd82a7286c
Step 4 : ENTRYPOINT /entrypoint.sh
 ---> Running in c4c00462c691
 ---> 62ad0d0bc850
Removing intermediate container c4c00462c691
Step 5 : CMD mysqld
 ---> Running in 1cd40a7e38f8
 ---> 16f80cf33da2
Removing intermediate container 1cd40a7e38f8
Successfully built 16f80cf33da2
```

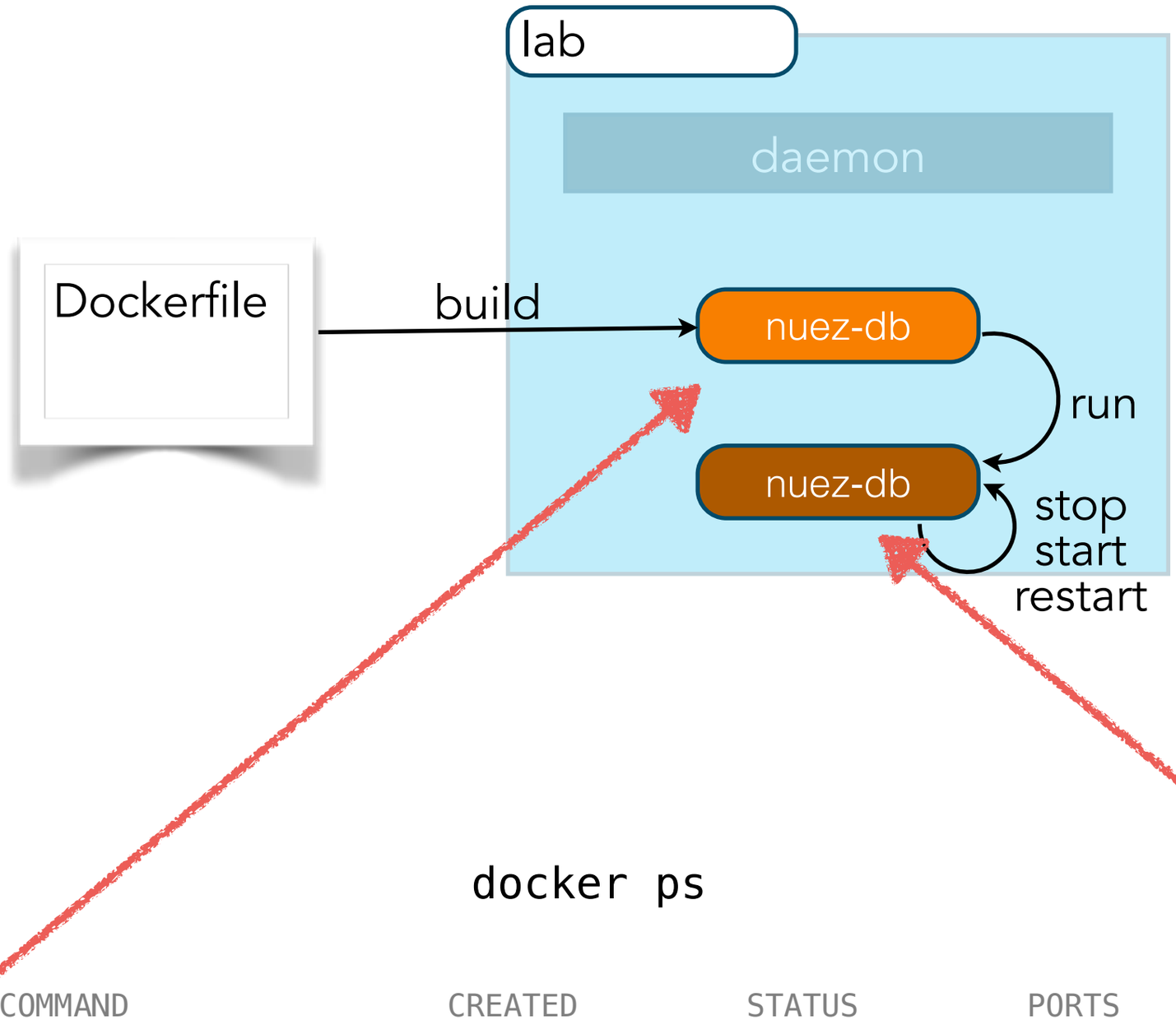



```
docker images
```

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nuez-db             latest              16f80cf33da2        8 minutes ago       214.4 MB
tomcat              7.0-jre7            c6838f52cb84        4 days ago          347.7 MB
mysql               5.5.45              0da0b10c6fd8        5 weeks ago         213.5 MB
```

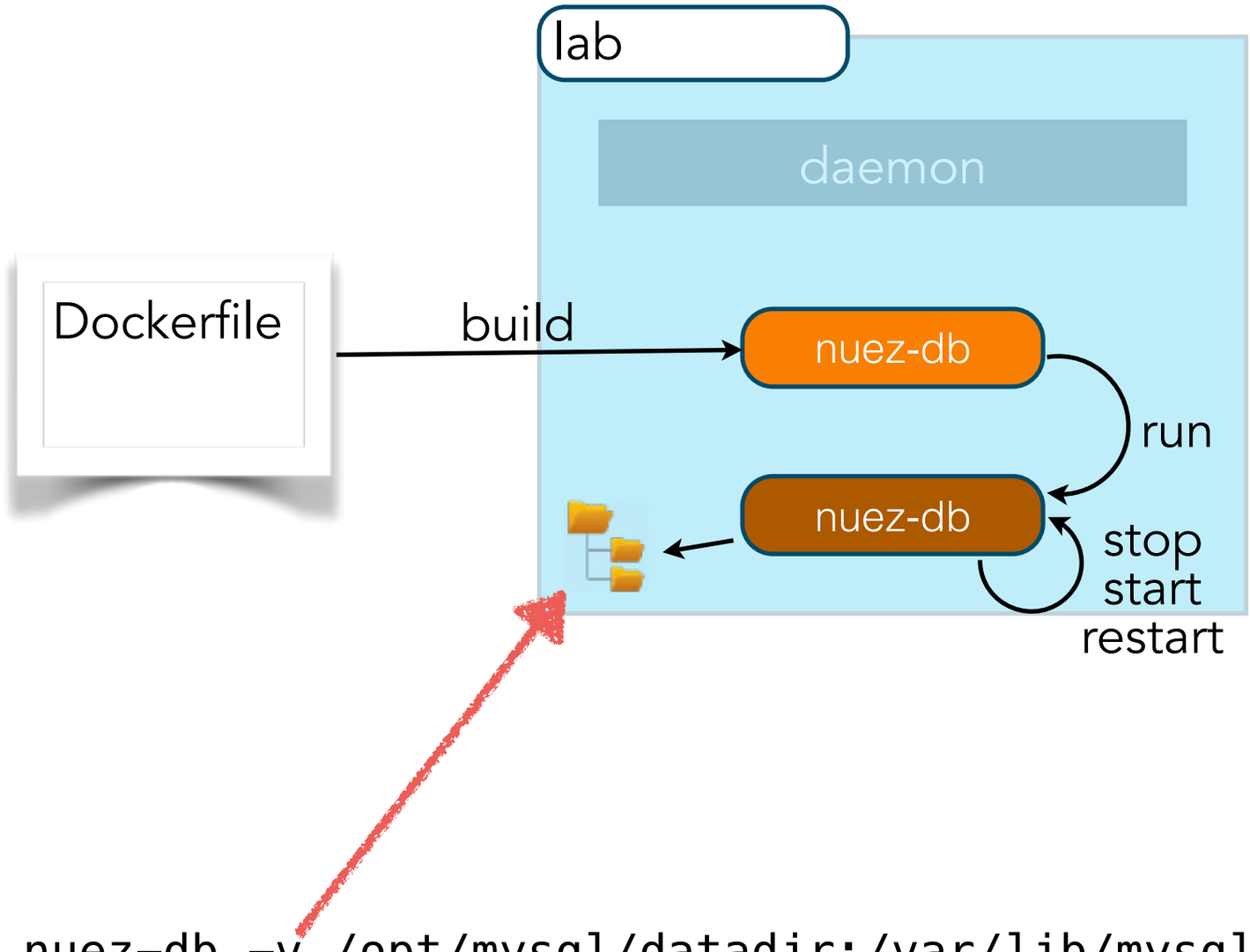



```
docker run --name nuez-db -e MYSQL_USER=nuez-app -e MYSQL_PASSWORD=nuez+1
  -e MYSQL_DATABASE=nuez -e MYSQL_ROOT_PASSWORD=root+1 -p 3306:3306 -d nuez-db

640a1bf669cc97763d51bc28865a5115a103e953336f8a7e90d50d61a108d0ec
```



docker ps

```
CONTAINER ID  IMAGE     COMMAND                  CREATED         STATUS         PORTS                     NAMES
640a1bf669cc  nuez-db   "/entrypoint.sh mysql"   11 minutes ago  Up 11 minutes  0.0.0.0:3306->3306/tcp    nuez-db
```



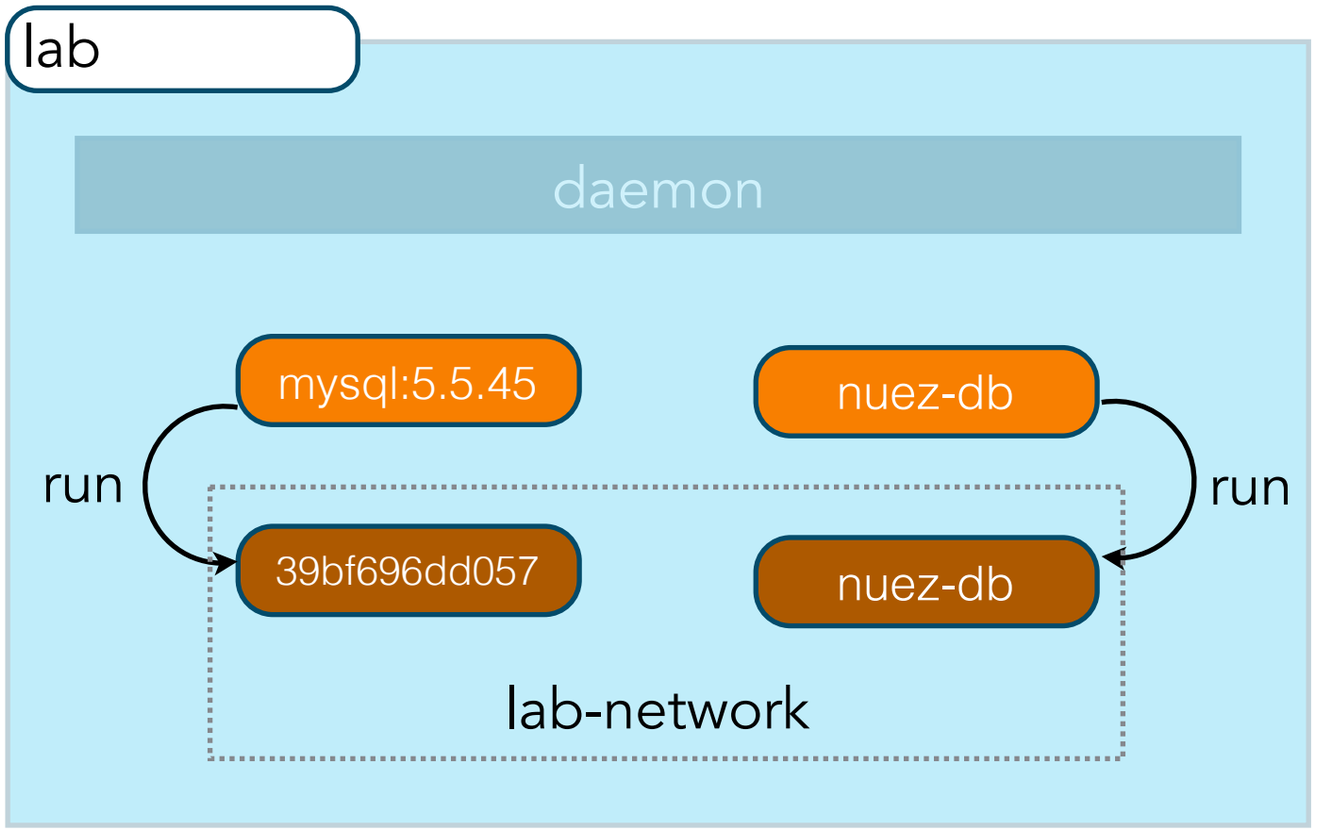

```
docker run --name nuez-db -v /opt/mysql/datadir:/var/lib/mysql \
 -e MYSQL_USER=nuez-app -e MYSQL_PASSWORD=nuez+1 \
 -e MYSQL_DATABASE=nuez -e MYSQL_ROOT_PASSWORD=root+1 -p 3306:3306 -d nuez-db
```

```
640a1bf669cc97763d51bc28865a5115a103e953336f8a7e90d50d61a108d0ec
```

Lab 4

1. Create a nuez-db image with MySQL
2. Run nuez-db container as a daemon

Network Containers


lab
daemon
mysql:5.5.45
nuez-db
run
run
39bf696dd057
nuez-db
lab-network

```
docker network ls
```

```
NETWORK ID          NAME                DRIVER              SCOPE
8d5ed05e81c1        bridge              bridge              local
acc72cedae53        host                host                local
505f6ef9f2ba        none                null                local
```

```
docker network create lab-network
```

```
218e86df00bad1598e9c1c05baa08c66b5a7fff2eeb3403fee5ed431d7a4eb0f
```

```
docker network ls
```

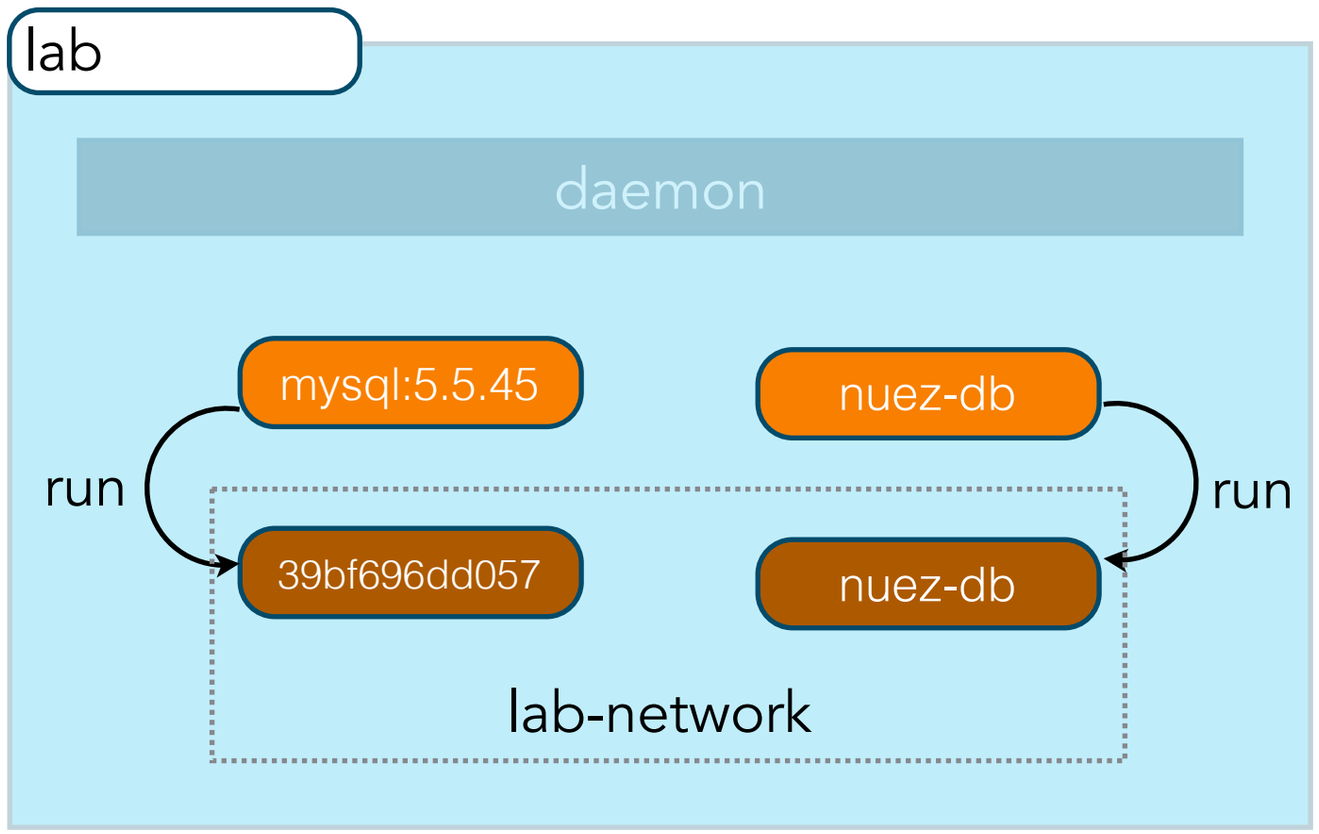
```
NETWORK ID          NAME                DRIVER              SCOPE
8d5ed05e81c1        bridge              bridge              local
acc72cedae53        host                host                local
218e86df00ba        lab-network         bridge              local
505f6ef9f2ba        none                null                local
```

```
docker network connect lab-network nuez-db
```

```
mysql -h 192.168.99.102 -u root -proot+1 nuez
```

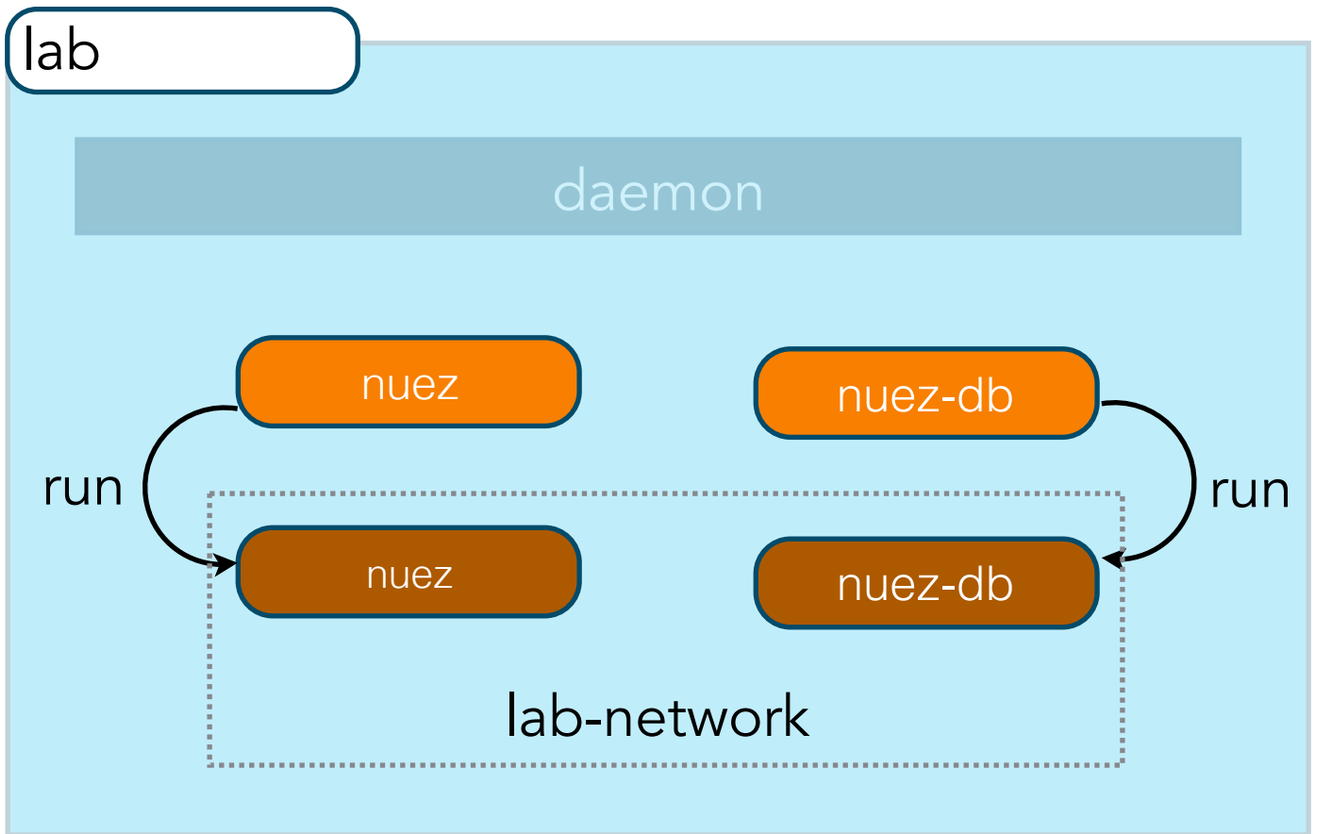
or

```
docker run -it --network lab-network --rm mysql:5.5.45 bash
mysql -h nuez-db -u root -p neuz
```

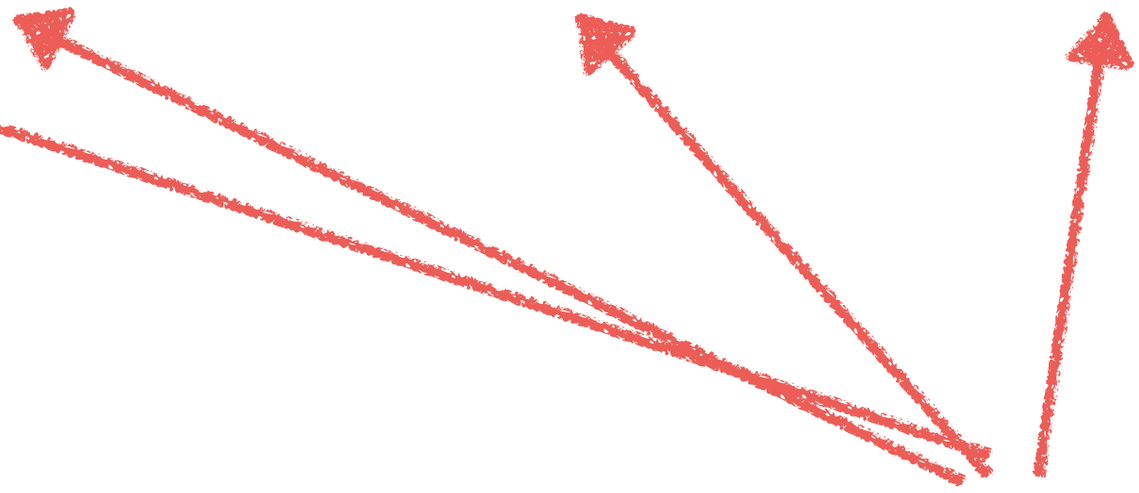


Lab 5

1. Run a mysql command-line in a container and connect to the nuez-db.


lab
daemon
nuez
nuez-db
run
run
nuez
nuez-db
lab-network

```
// remove for brevity

docker {
      dataSource {
          dbCreate = "update"
          url = "jdbc:mysql://${System.getenv()['DB_SERVER']}:${System.getenv()['DB_PORT']}/${System.getenv()['DB_DB']}"
          driverClassName = "com.mysql.jdbc.Driver"
          username = System.getenv()['DB_USER']
          password = System.getenv()[‘DB_PASSWORD']
          pooled = true
          properties {
             maxActive = -1
             minEvictableIdleTimeMillis=1800000
             timeBetweenEvictionRunsMillis=1800000
             numTestsPerEvictionRun=3
             testOnBorrow=true
             testWhileIdle=true
             testOnReturn=true
             validationQuery="SELECT 1"
          }
      }
  }
// remove for brevity
```



Dockerfile

```
FROM tomcat:7.0-jre7

RUN rm -rf $CATALINA_HOME/webapps/ROOT
RUN rm -rf $CATALINA_HOME/webapps/ROOT.war
RUN rm -rf $CATALINA_HOME/webapps/docs
RUN rm -rf $CATALINA_HOME/webapps/examples
RUN rm -rf $CATALINA_HOME/webapps/manager
RUN rm -rf $CATALINA_HOME/webapps/host-manager

ENV CATALINA_OPTS -Dgrails.env=docker

ADD https://s3.amazonaws.com/cmj-presentations/docker-for-devs/nuez.war $CATALINA_HOME/webapps/ROOT.war

CMD ["catalina.sh", "run"]
```

```
docker build -t nuez .
```

Dockerfile

```
FROM tomcat:7.0-jre7

RUN rm -rf $CATALINA_HOME/webapps/ROOT
RUN rm -rf $CATALINA_HOME/webapps/ROOT.war
RUN rm -rf $CATALINA_HOME/webapps/docs
RUN rm -rf $CATALINA_HOME/webapps/examples
RUN rm -rf $CATALINA_HOME/webapps/manager
RUN rm -rf $CATALINA_HOME/webapps/host-manager

ENV CATALINA_OPTS -Dgrails.env=docker

ADD nuez.war $CATALINA_HOME/webapps/ROOT.war

CMD ["catalina.sh", "run"]
```

```
docker build -t nuez .
```

```
docker run --name nuez --network lab-network -d  \
  -e DB_SERVER=nuez-db -e DB_PORT=3306 -e DB_DB=nuez \
-e DB_USER=nuez-app -e DB_PASSWORD=nuez+1 \
-p 80:8080 nuez
```

Nuez | Home

192.168.99.101

aws regatta iqity manifest willowwood judd codemash hadoop devtools devops old clients

NUEZ

The blog about anything....really...Anything!

Home All Posts About Login

Ads

Buy Stuff Here

And more stuff here

If you like stuff, you'll like this stuff...

More stuff here.

But I spent all my money on stuff.

Hello!

Welcome to the Nuez blog. Please feel free to login and blog about any topic that you want to talk about. Enjoy!

Learn more »

Login

192.168.99.101/login/auth

aws | regatta | iqity | manifest | willowwood | judd | codemash | hadoop | devtools | devops | old clients

NUEZ

The blog about anything....really...Anything!

Home All Posts About Login

Please Login

Username: blogger

Password: ••••••••

Remember me

Login

blogger

1234abcd

```
docker exec -it nuez bash
```

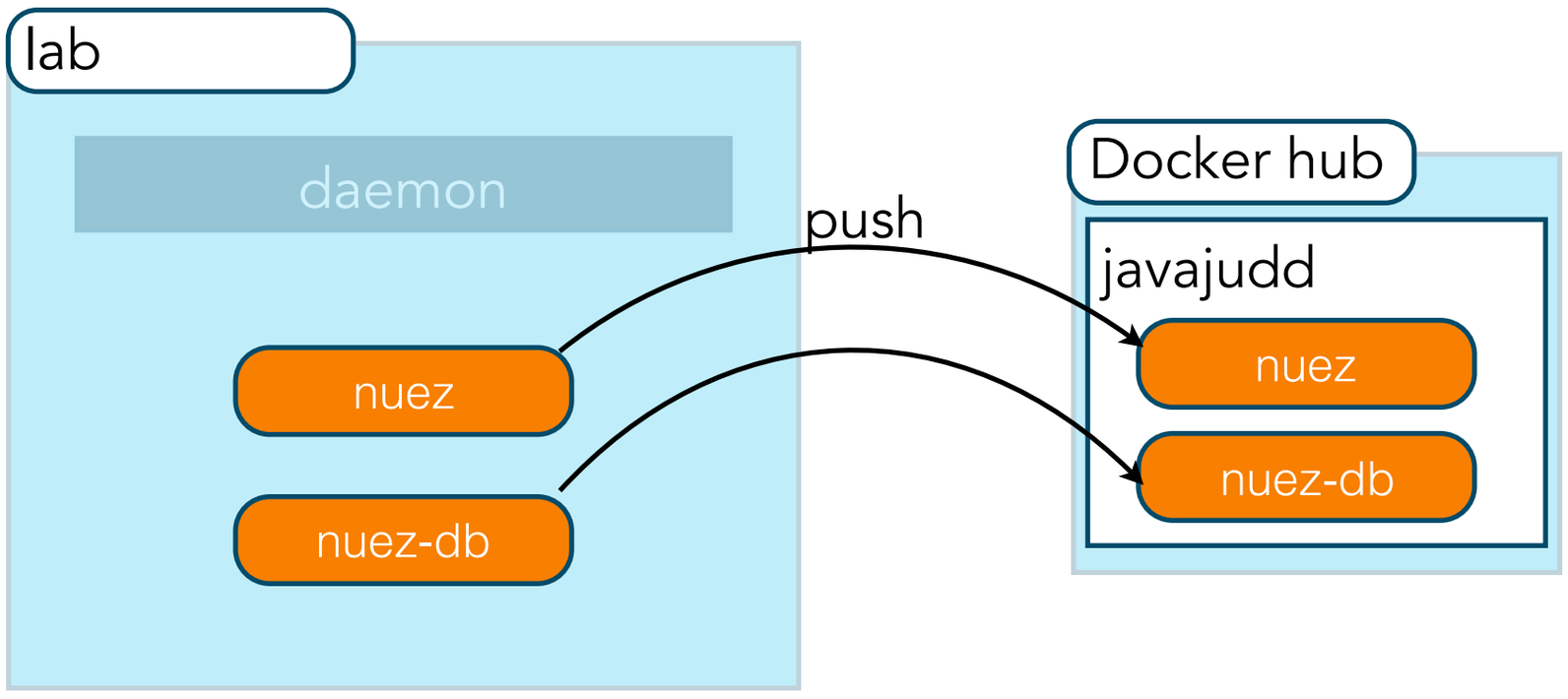
```
docker logs nuez
```

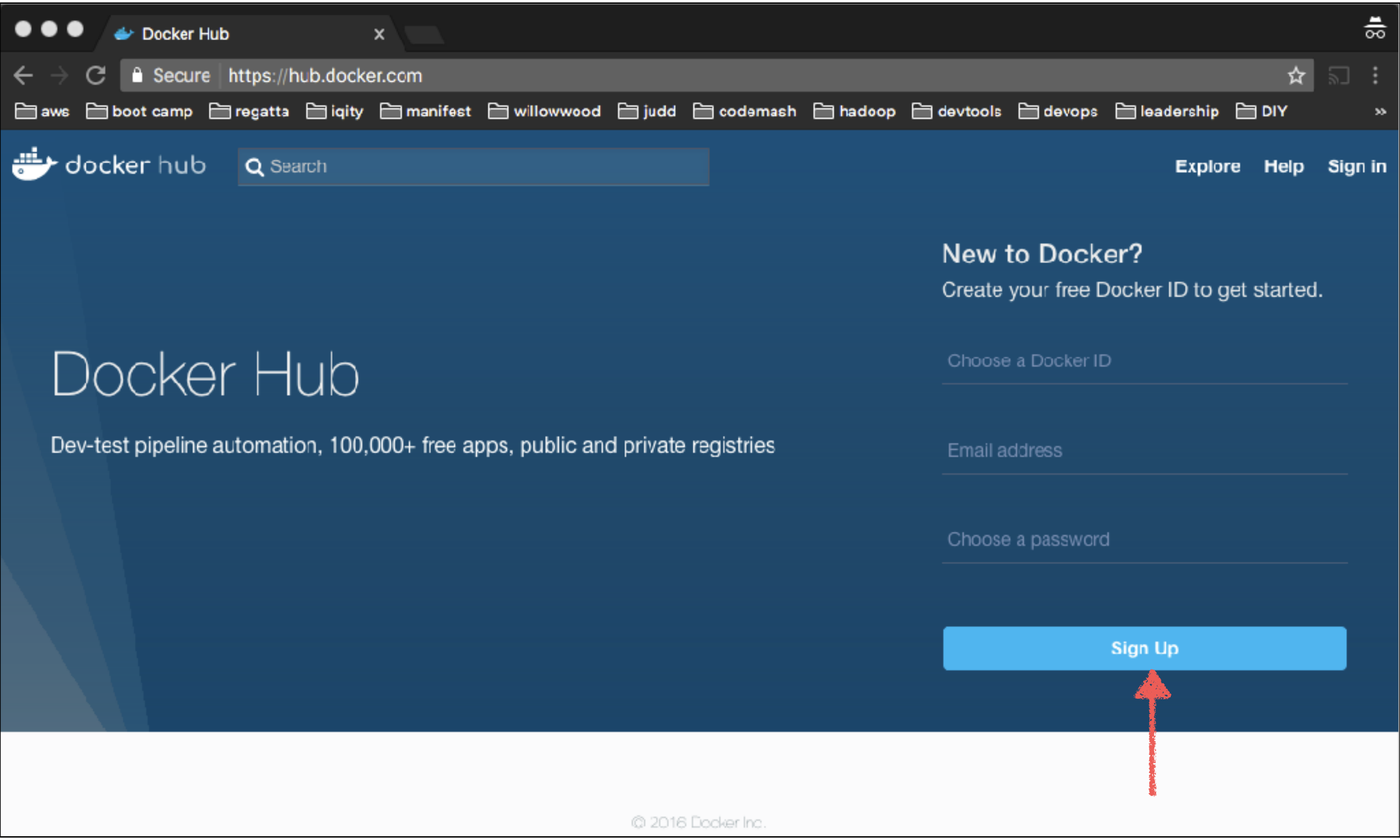
```
docker diff nuez
```

Lab 6

1. Create a nuez image containing Tomcat 7 and Java 7
2. Run the container
3. Test the container
4. Look at the Tomcat log files

Share Images


lab
daemon
nuez
nuez-db
push
Docker hub
javajudd
nuez
nuez-db



https://hub.docker.com/

```
docker login

docker build -t javajudd/nuez-db:1.0 .
docker push javajudd/nuez-db

docker build -t javajudd/nuez:1.0 .
docker push javajudd/nuez
```

Dashboard Explore Organizations Search Create javajudd

PUBLIC REPOSITORY

javajudd/nuez-db ☆

Last pushed: a few seconds ago

Repo Info Tags Collaborators Webhooks Settings

Short Description

Mysql container for hosting the nuez database.

Full Description

Full description is empty for this repo.

Docker Pull Command

```
docker pull javajudd/nuez-db
```

Owner

javajudd

Comments (0)

Add Comment

Dashboard Explore Organizations Search Create javajudd

PUBLIC REPOSITORY

javajudd/nuez-db ☆

Last pushed: a minute ago

Repo Info | Tags | Collaborators | Webhooks | Settings

Tags

Tag	Size
latest	70 MB
1.0	70 MB

Docker Pull Command

```
docker pull javajudd/nuez-db
```

Owner

javajudd

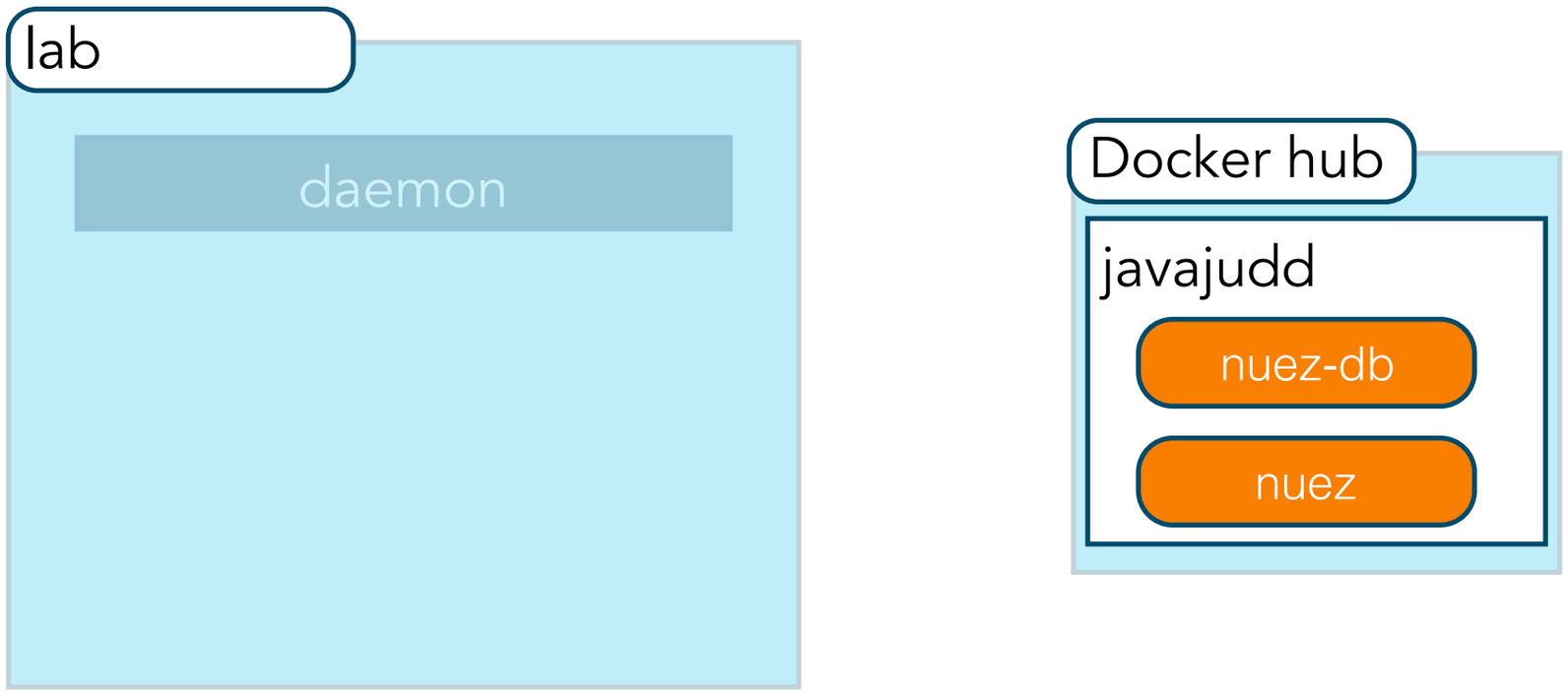
Comments (0)

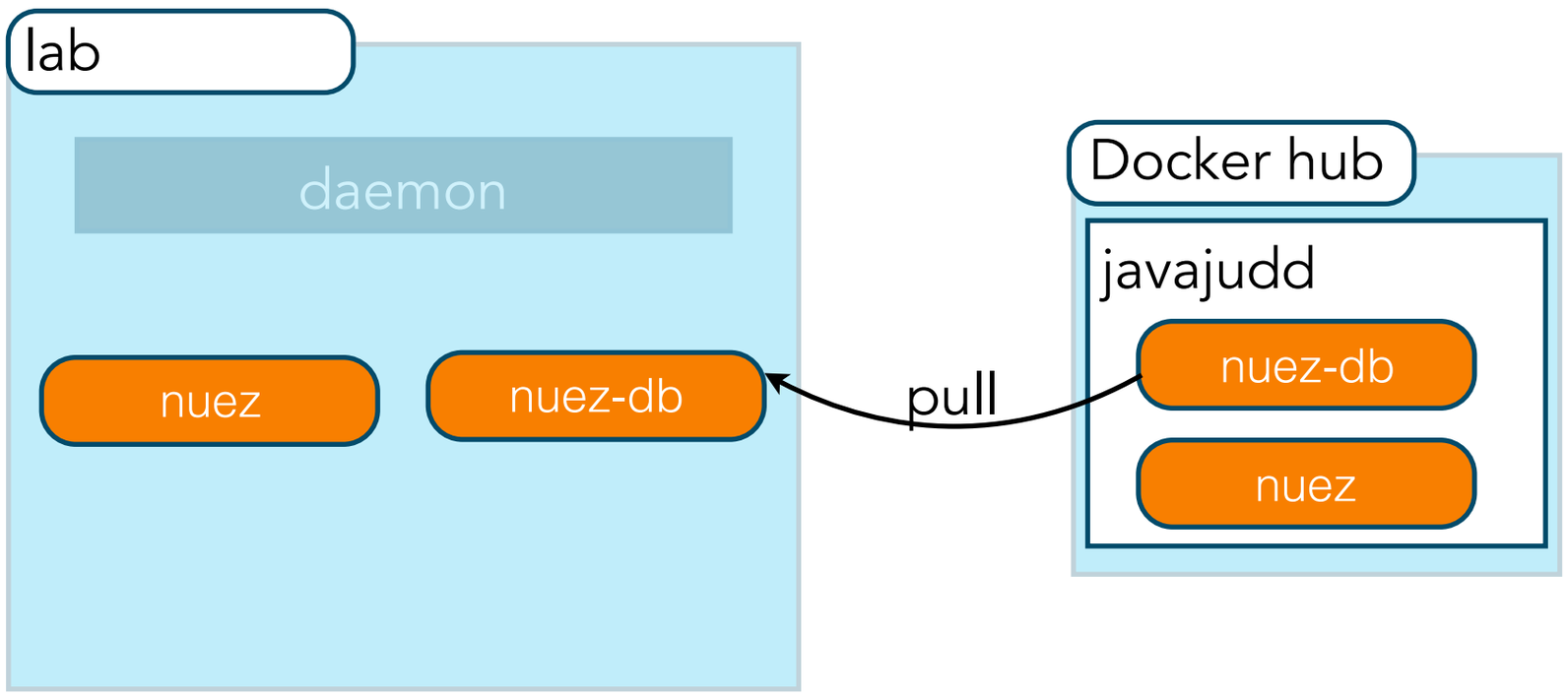
Add Comment

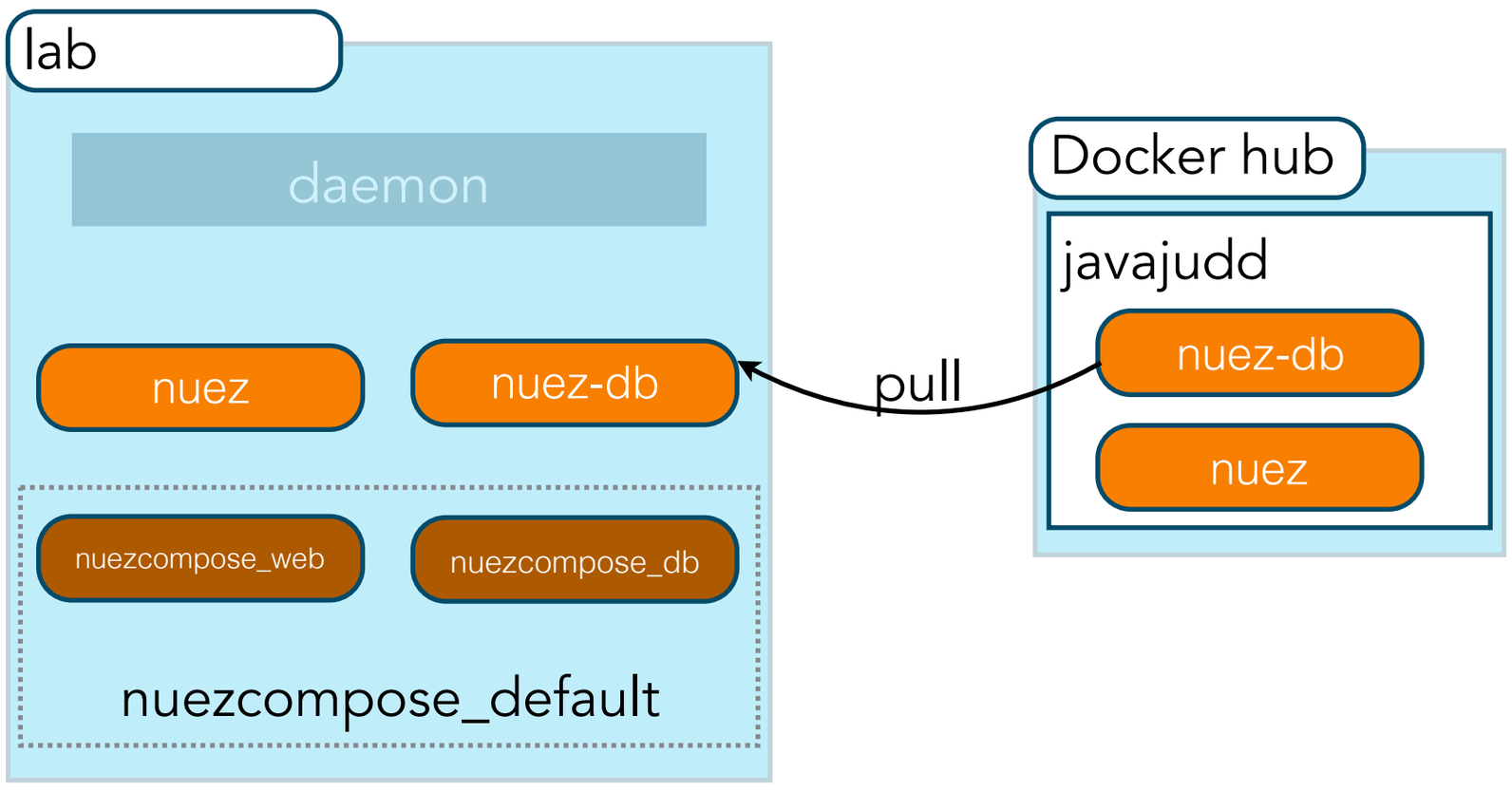
Lab 7

1. Push nuez and nuez-db docker containers to your repository

Composing


lab
daemon
Docker hub
javajudd
nuez-db
nuez


lab
daemon
nuez
nuez-db
pull
Docker hub
javajudd
nuez-db
nuez


lab
daemon
nuez
nuez-db
nuezcompose_web
nuezcompose_db
nuezcompose_default
pull
Docker hub
javajudd
nuez-db
nuez

docker-compose.yml

```
version: "3"
services:
  db:
    image: javajudd/nuez-db:1.0
    ports:
      - "3306:3306"
    environment:
       MYSQL_USER: ${DB_USER}
       MYSQL_PASSWORD: ${DB_PASSWORD}
       MYSQL_DATABASE: ${DB_DB}
       MYSQL_ROOT_PASSWORD: ${DB_ROOT_PASS}
  web:
    image: javajudd/nuez:1.0
    ports:
      - "80:8080"
    environment:
       DB_USER: ${DB_USER}
       DB_PASSWORD: ${DB_PASSWORD}
       DB_DB: ${DB_DB}
       DB_SERVER: db
       DB_PORT: 3306
```

.env

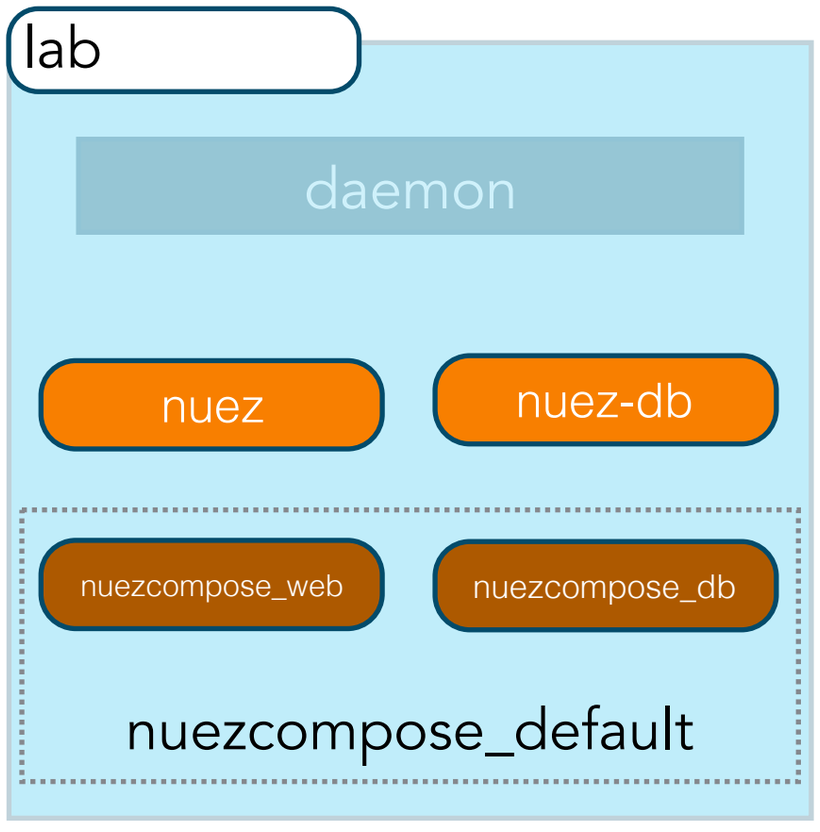
```
DB_USER=nuez-app
DB_PASSWORD=nuez+1
DB_DB=nuez
DB_ROOT_PASS=root+1
```

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

```
Creating network "nuezcompose_default" with the default driver
Creating nuezcompose_web_1 ... done
Creating nuezcompose_db_1  ... done
```

docker ps

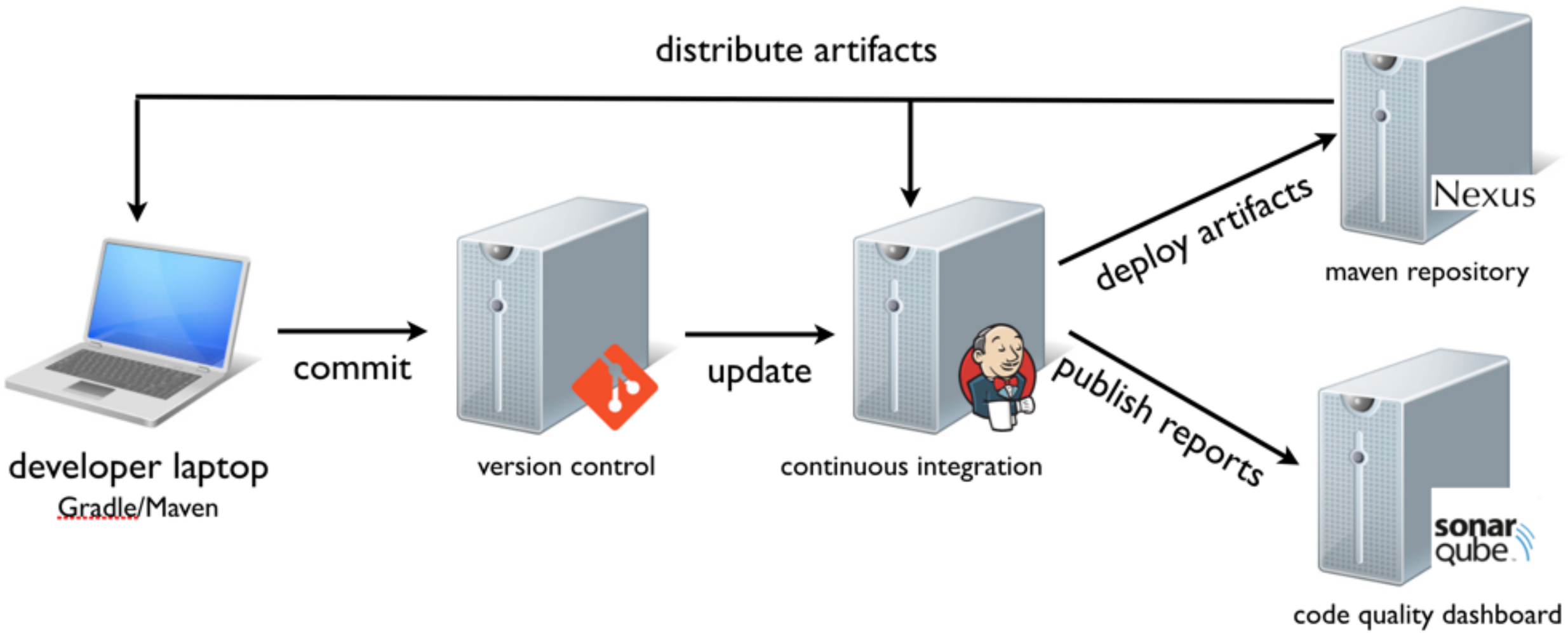
```
CONTAINER ID        IMAGE                     COMMAND                  PORTS                    NAMES
59e9a2e827e0        javajudd/nuez:1.0         "catalina.sh run"        0.0.0.0:80->8080/tcp     nuezcompose_web_1
8eafecf16a37        javajudd/nuez-db:1.0      "/entrypoint.sh mysq…"   0.0.0.0:3306->3306/tcp   nuezcompose_db_1
```



```
docker-compose down
```

```
Stopping nuezcompose_web_1 ... done
Stopping nuezcompose_db_1  ... done
Removing nuezcompose_web_1 ... done
Removing nuezcompose_db_1  ... done
Removing network nuezcompose_default
```

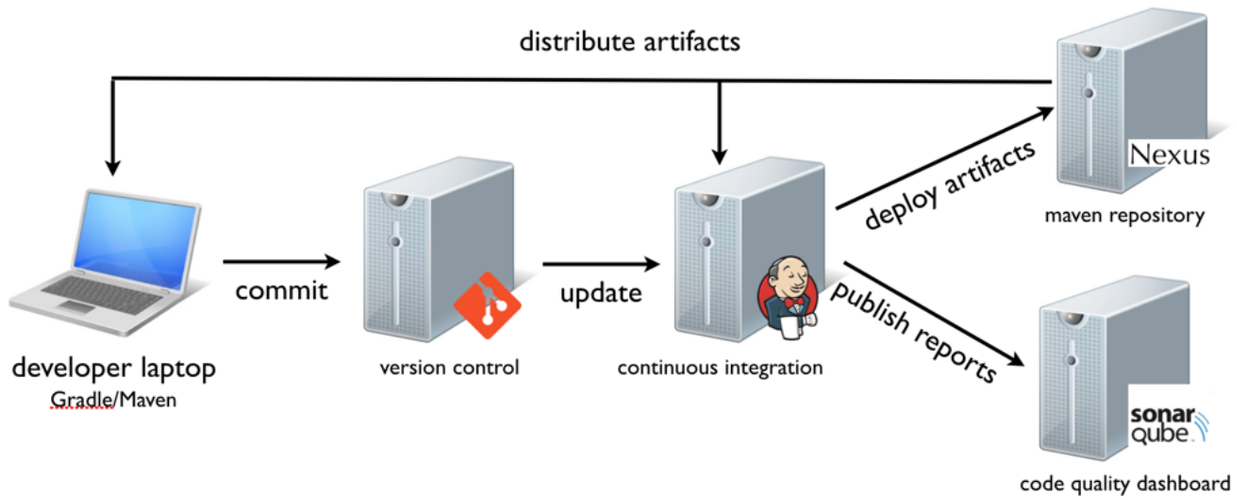
Ultimate Enterprise Java Build System



https://github.com/cjudd/ultimateenterprisejavabuildsystem/blob/master/docker-compose.yml

Ultimate Enterprise Java Build System

```
version: '2'
services:

  jenkins:
    image: jenkins
    ports:
     - "9001:8080"
     - "50000:50000"
    volumes:
     - /var/jenkins_home

  nexus:
    image: sonatype/nexus
    ports:
     - "8081:8081"
    volumes:
     - /sonatype-work

postgres:
  image: postgres
  volumes:
   - /var/lib/postgressql/data
  environment:
   - POSTGRES_USER=sonar
   - POSTGRES_PASSWORD=sonar
   - POSTGRES_DB=sonar

sonarqube:
  image: sonarqube
  links:
   - postgres
  ports:
   - "9000:9000"
   - "9092:9092"
  volumes:
   - /opt/sonarqube
  environment:
   - SONARQUBE_JDBC_USERNAME=sonar
   - SONARQUBE_JDBC_PASSWORD=sonar
   - SONARQUBE_JDBC_URL=jdbc:postgresql://postgres/sonar
```



Lab 8

1. Compose nuez and nuez-db

Docker & Java

https://hub.docker.com/_/java/

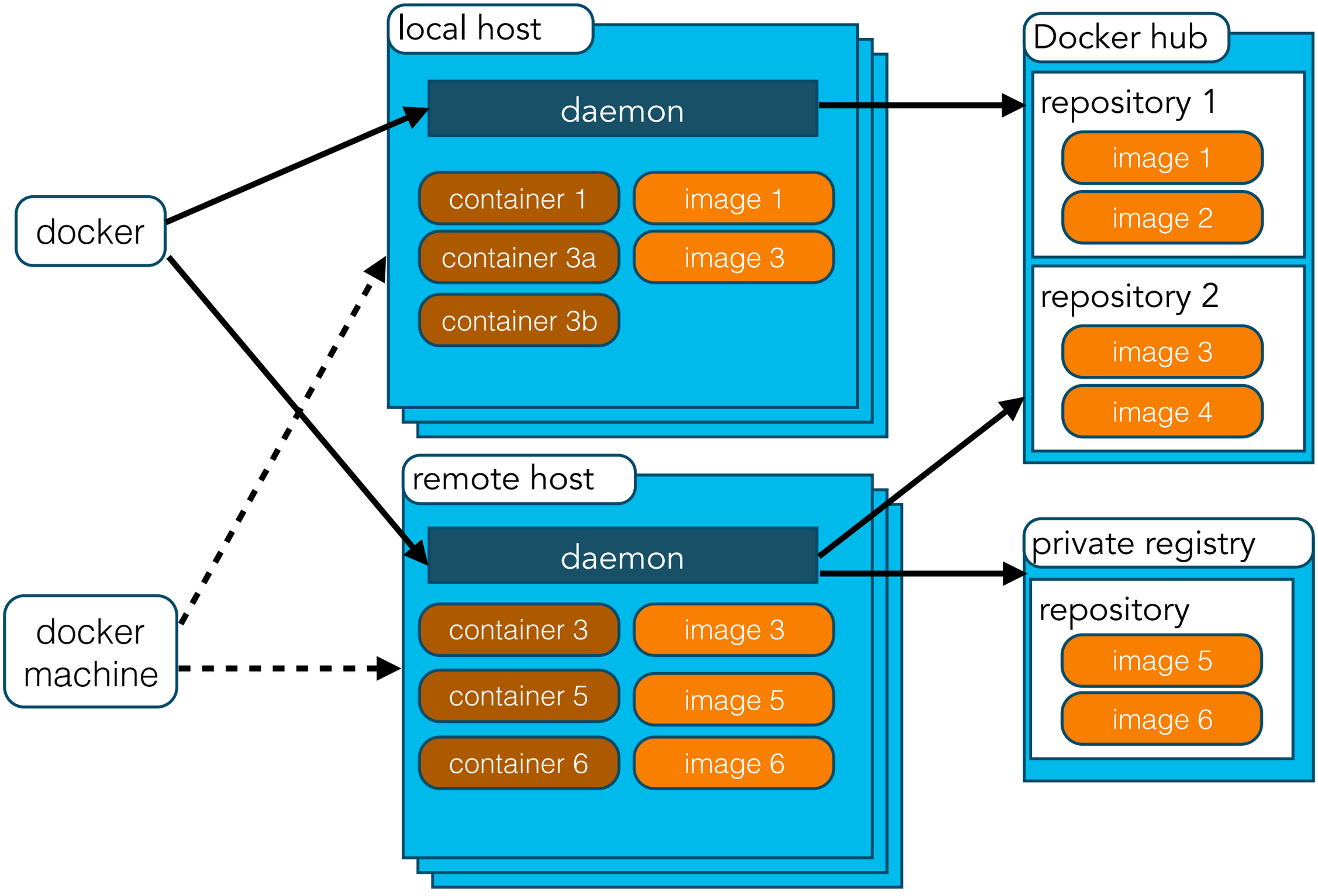
https://hub.docker.com/_/openjdk/

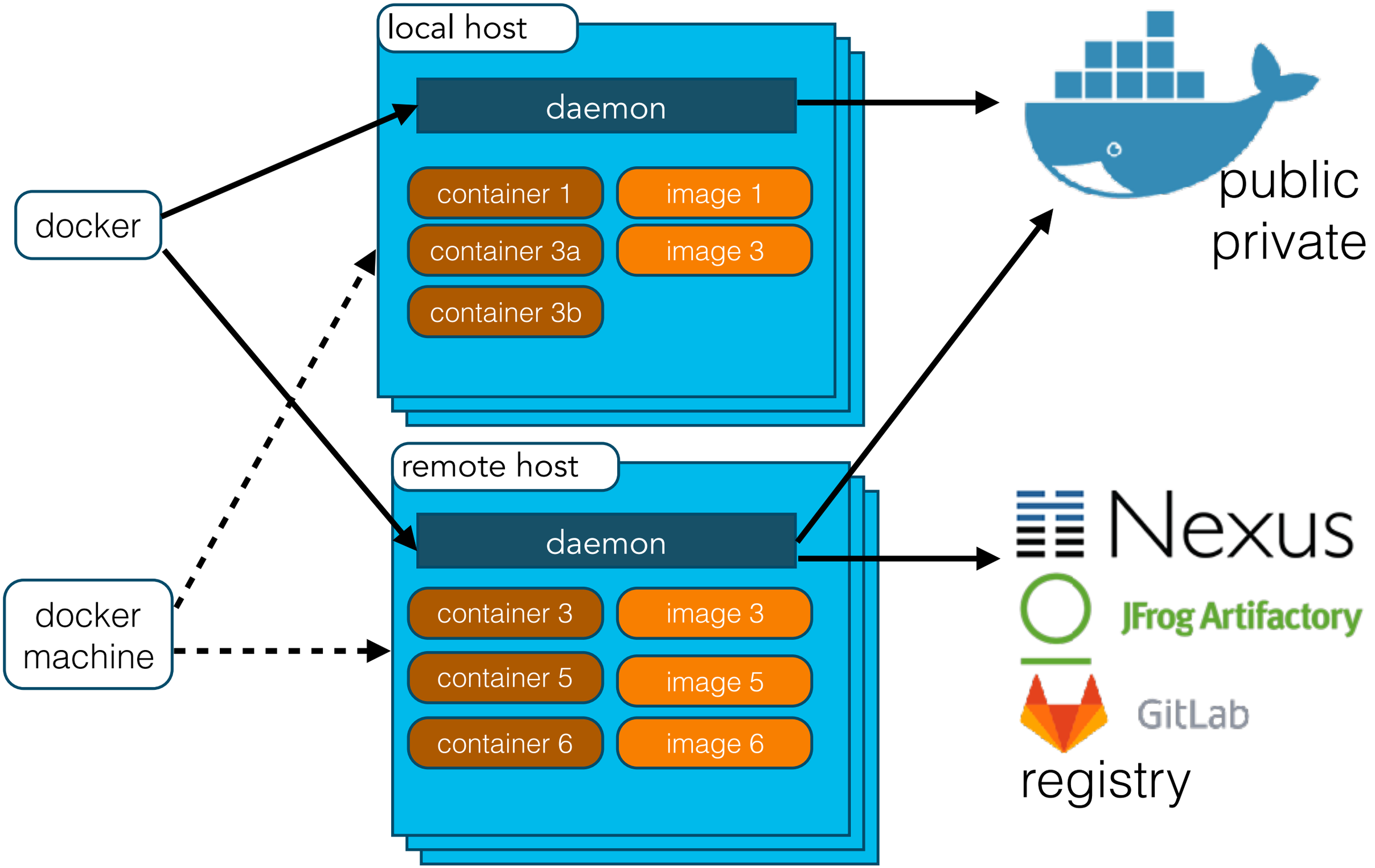
https://store.docker.com/images/oracle-serverjre-8

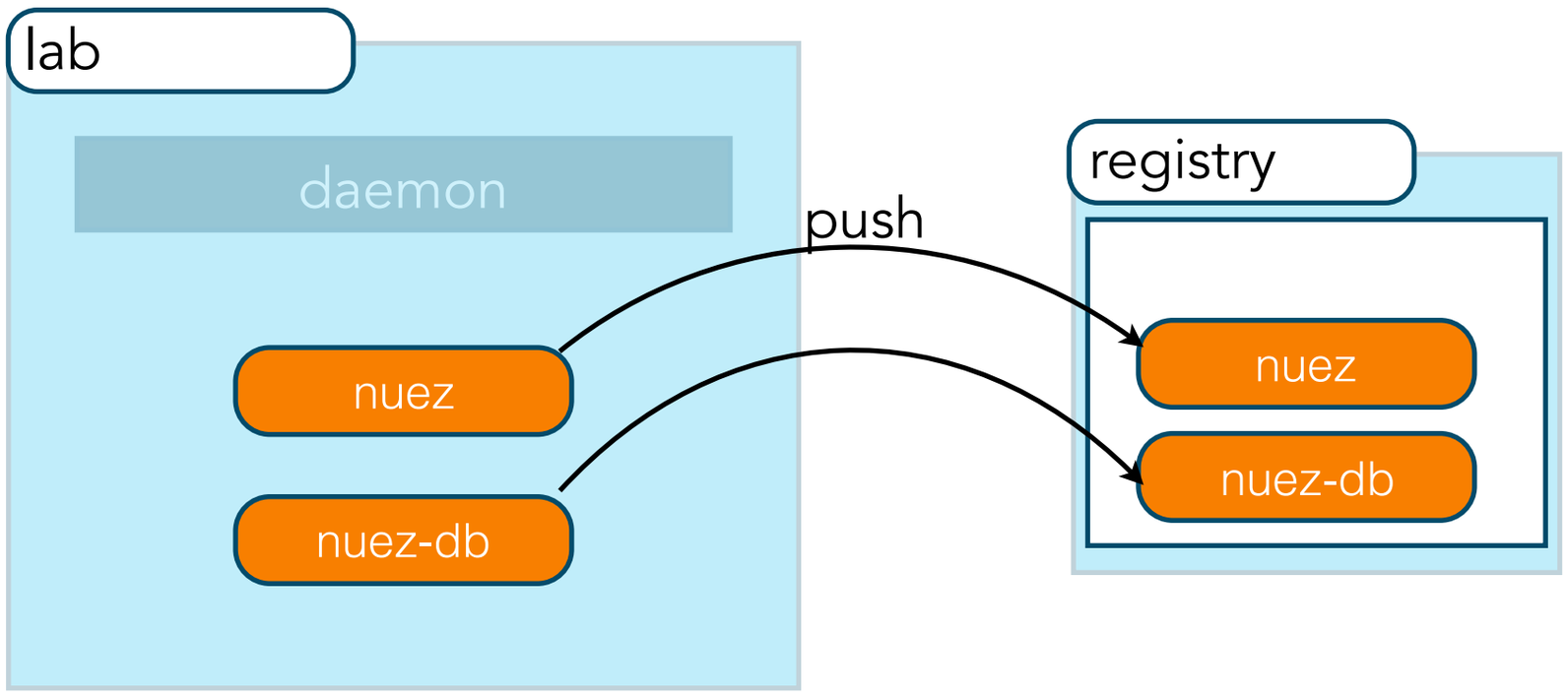
```
docker run -it --rm openjdk:9 jshell
```

```
Mar 13, 2018 6:42:43 PM java.util.prefs.FileSystemPreferences$1 run
INFO: Created user preferences directory.
|  Welcome to JShell -- Version 9.0.1
|  For an introduction type: /help intro

jshell> 3 + 4
$1 ==> 7

jshell> /exit
|  Goodbye
```

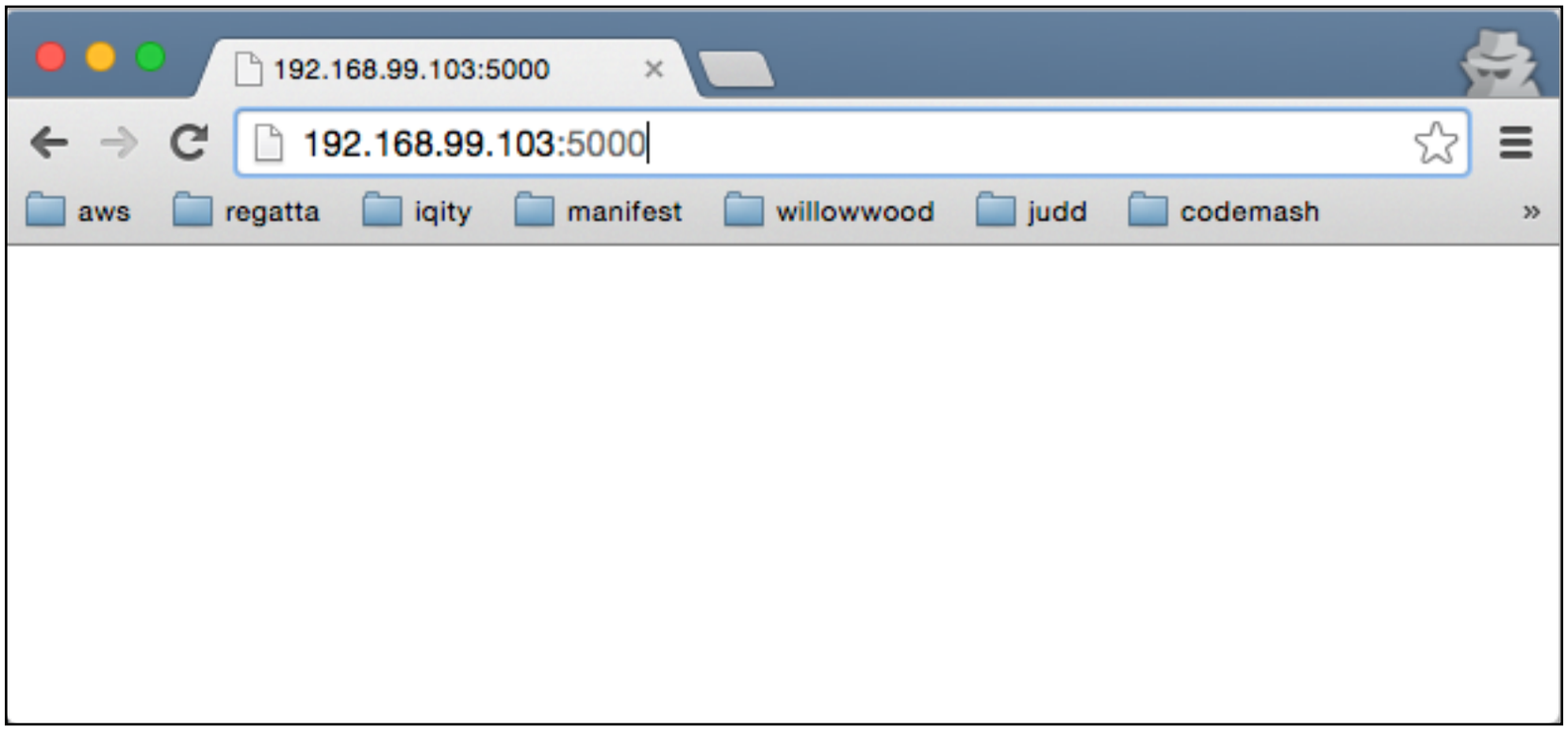
```
docker run --memory   >   java -Xmx
```


local host
daemon
container 1
image 1
container 3a
image 3
container 3b
docker
Docker hub
repository 1
image 1
image 2
repository 2
image 3
image 4
remote host
daemon
container 3
image 3
container 5
image 5
container 6
image 6
docker
machine
private registry
repository
image 5
image 6


local host
daemon
container 1
image 1
container 3a
image 3
container 3b
docker
public
private
remote host
daemon
container 3
image 3
container 5
image 5
container 6
image 6
docker
machine
Nexus
JFrog Artifactory
GitLab
registry


lab
daemon
nuez
nuez-db
push
registry
nuez
nuez-db

```
docker run -d -p 5000:5000 --name registry registry:2.2.0
```



```
docker tag nuez-db localhost:5000/nuez-db
docker push localhost:5000/nuez-db
```

```
docker tag nuez localhost:5000/nuez
docker push localhost:5000/nuez
```

Install
Docker Fundamentals
Use Docker
Manage image repositories
Extend Docker
Command and API references
Docker run reference
Dockerfile reference
Remote API client libraries
Using the command line
docker.io accounts API
Docker Remote API
Docker Hub

Docker Registry HTTP API V2

Introduction

The *Docker Registry HTTP API* is the protocol to facilitate distribution of images to the docker engine. It interacts with instances of the docker registry, which is a service to manage information about docker images and enable their distribution. The specification covers the operation of version 2 of this API, known as *Docker Registry HTTP API V2*.

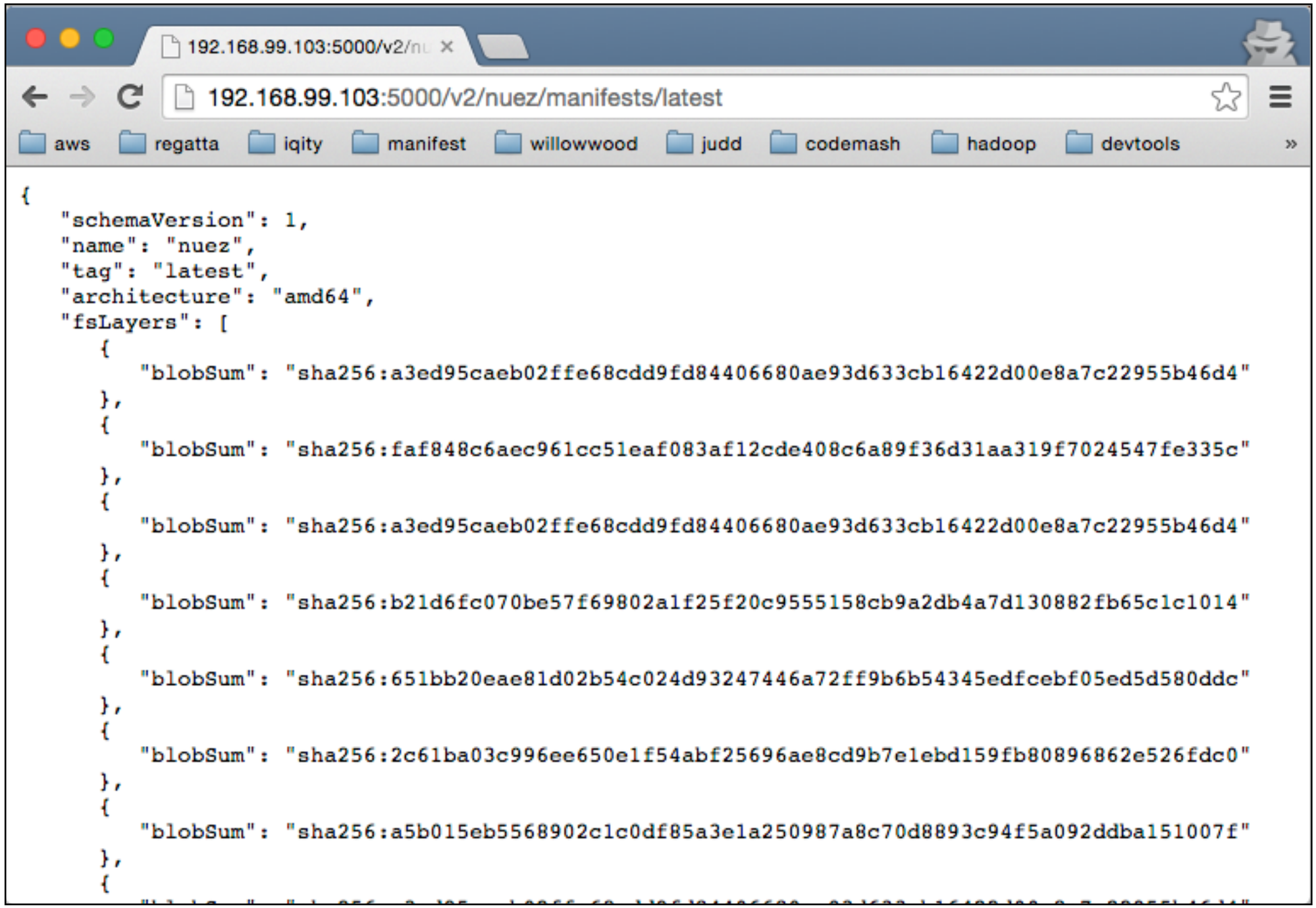
While the V1 registry protocol is usable, there are several problems with the architecture that have led to this new version. The main driver of this specification these changes to the docker the image format, covered in docker/docker#8093. The new, self-contained image manifest simplifies image definition and improves security. This specification will build on that work, leveraging new properties of the manifest format to improve performance, reduce bandwidth usage and decrease the likelihood of backend corruption.

For relevant details and history leading up to this specification, please see the following issues:

On this page:
Docker Registry HTTP API V2
Introduction
Scope
Use Cases
Changes
Overview
Errors
API Version Check
Content Digests
Pulling An Image
Pushing An Image
Deleting a Layer
Listing
Repositories
Listing Image Tags
Deleting an Image
Detail
Errors
Base
Tags
Manifest
Blob
Initiate Blob
Upload
Blob Upload
Catalog

https://docs.docker.com/registry/spec/api/

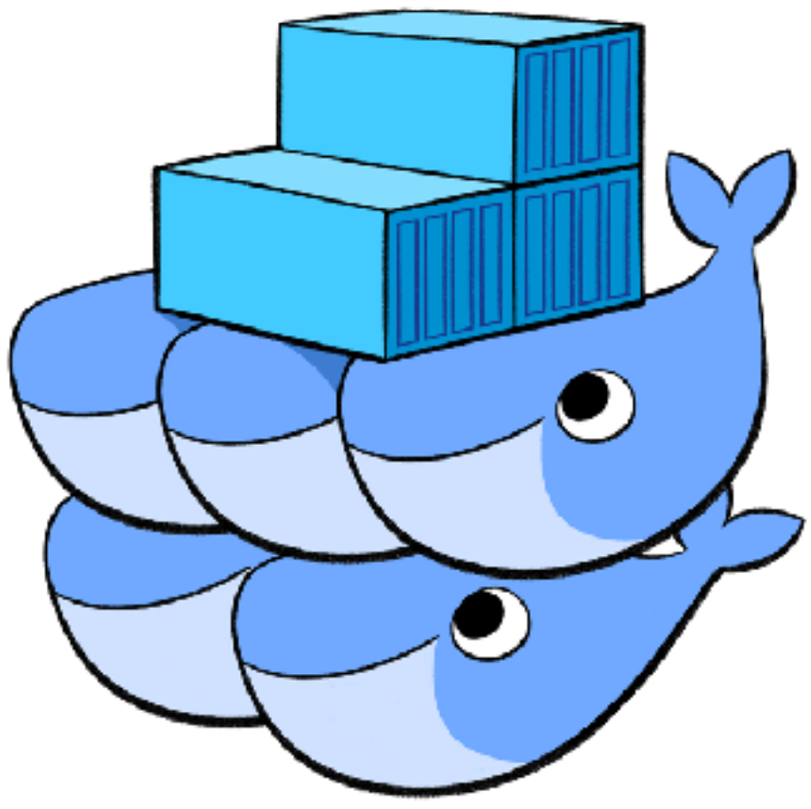
http://192.168.99.103:5000/v2/nuez/manifests/latest



```
docker pull localhost:5000/nuez-db
docker pull localhost:5000/nuez
```

Clustering

kubernetes



kubernetes

View Documentation

Production-Grade Container Orchestration

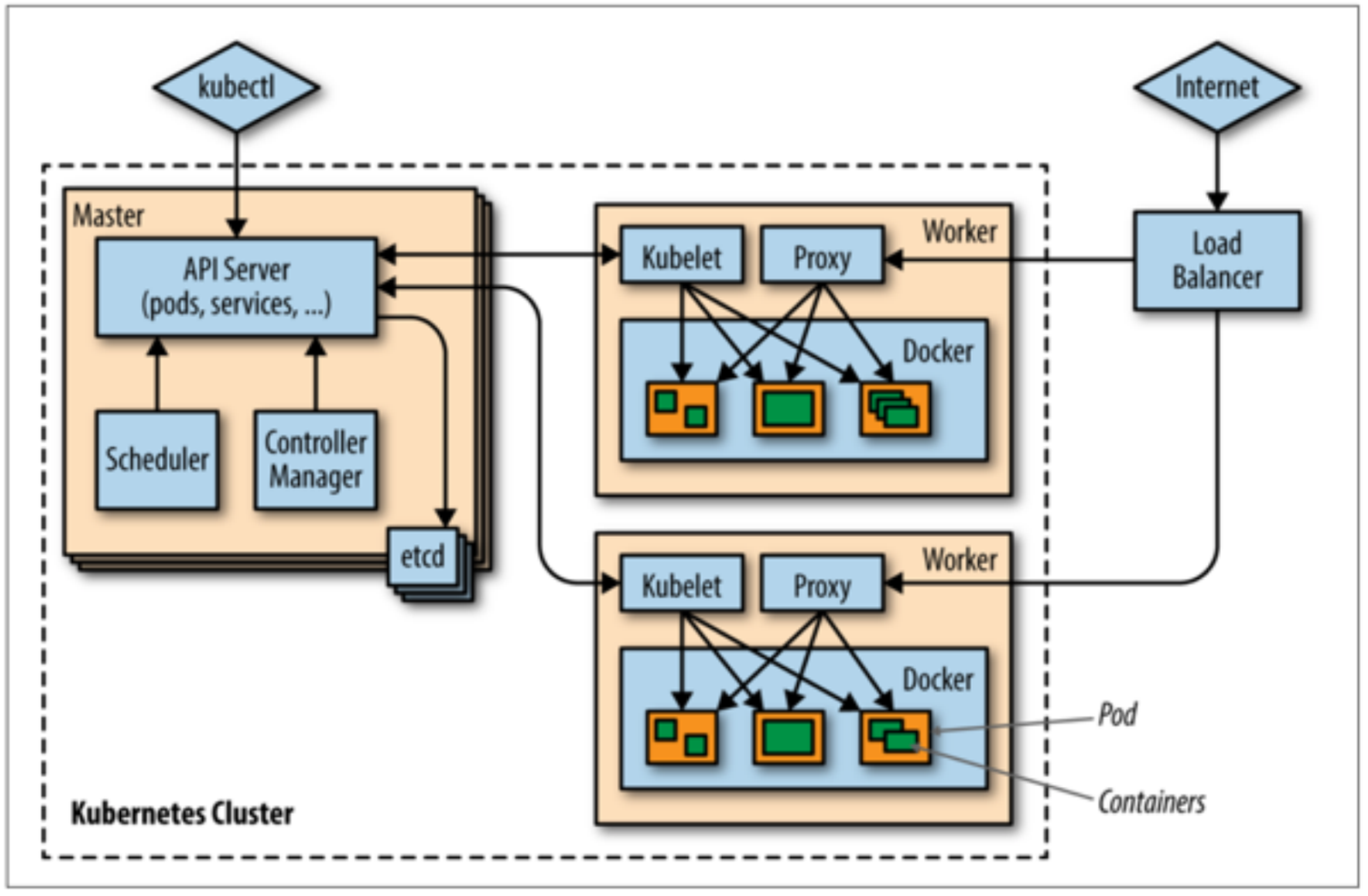
Automated container deployment, scaling, and management

Try Our Hello World

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.

http://kubernetes.io/


kubectl
Internet
Master
API Server
(pods, services, ...)
Scheduler
Controller
Manager
etcd
Worker
Kubelet
Proxy
Docker
Load
Balancer
Worker
Kubelet
Proxy
Docker
Pod
Containers
Kubernetes Cluster

- Install Hybervisor (xhyve, VirtualBox, VMWare, Hyper-V)
- Install kubectl
- Install Minikube

https://kubernetes.io/docs/tasks/tools/install-minikube/

```
minikube start
```

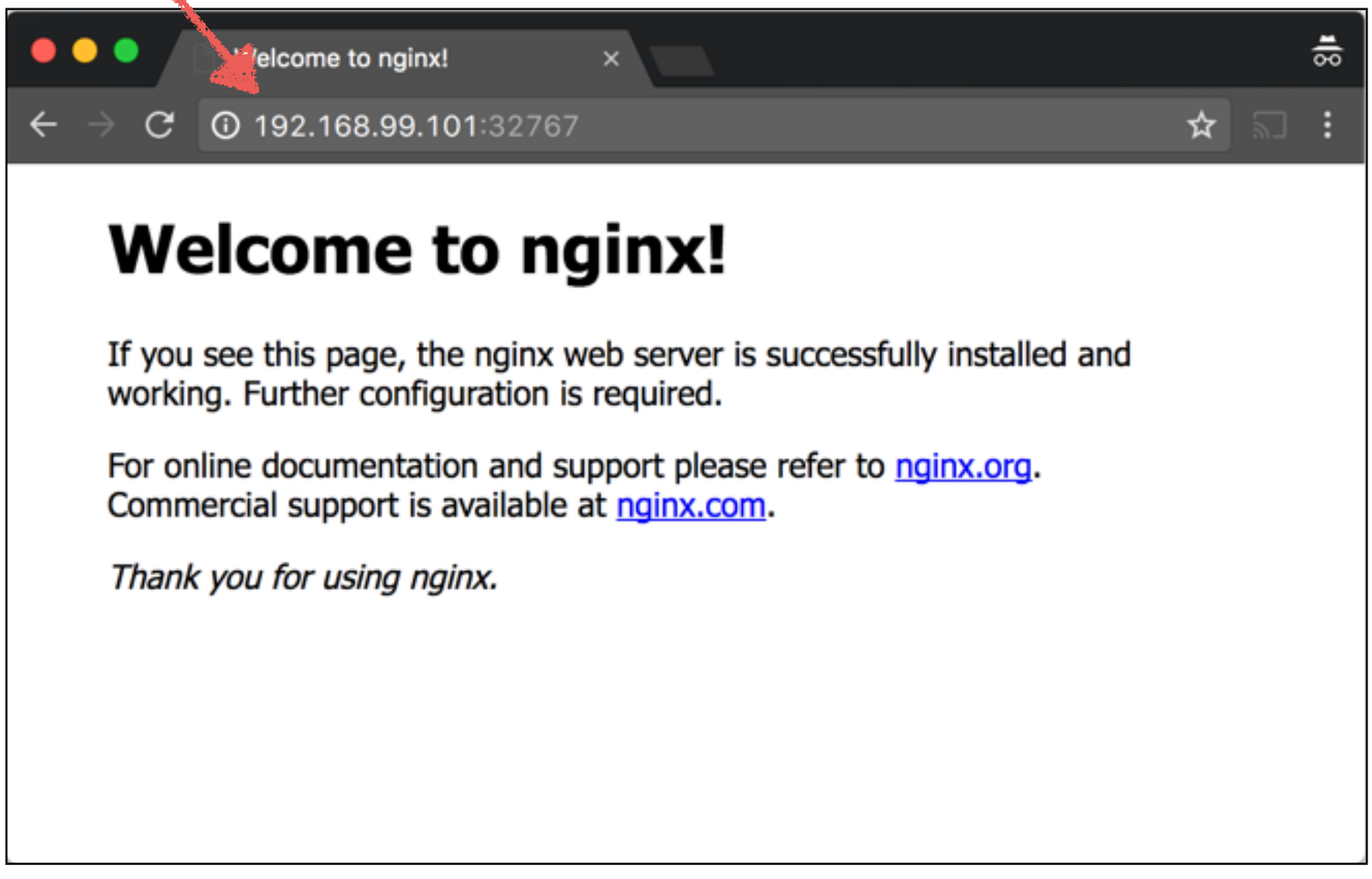
```
Starting local Kubernetes v1.6.4 cluster...
Starting VM...
Downloading Minikube ISO
 90.95 MB / 90.95 MB [============================================] 100.00% 0s
Moving files into cluster...
Setting up certs...
Starting cluster components...
Connecting to cluster...
Setting up kubeconfig...
Kubectl is now configured to use the cluster.
```

```
minikube ip
```

```
192.168.99.101
```

```
kubectl run lb-nginx --image=nginx --replicas=2 --port=80
deployment "lb-nginx" created

kubectl get pod
NAME                        READY     STATUS    RESTARTS   AGE
lb-nginx-4263643773-54l9b   1/1       Running   0          3m
lb-nginx-4263643773-ddknt   1/1       Running   0          3m


kubectl get deployment
NAME       DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
lb-nginx   2         2         2            2           4m

kubectl expose deployment lb-nginx --type=LoadBalancer
service "lb-nginx" exposed

kubectl get services
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes   10.0.0.1      <none>        443/TCP        9h
lb-nginx     10.0.0.161    <pending>     80:32767/TCP   4m
```

```
minikube service lb-nginx --url
http://192.168.99.101:32767
```



```
kubectl delete services lb-nginx
service "lb-nginx" deleted
kubectl delete deployment lb-nginx
deployment "lb-nginx" deleted

minikube stop
```

Cloud

DigitalOcean

https://www.digitalocean.com/?refcode=3c8ba4775cf9

Simple Pricing

All plans are standard with **solid state drives** (SSD).

MONTHLY HOURLY

Pricing in USD. Excludes any applicable tax.

$5/mo

512MB Memory

1 Core Processor

20GB SSD Disk

1TB Transfer

SIGN UP

$10/mo

Most Popular Plan

1GB Memory

1 Core Processor

30GB SSD Disk

2TB Transfer

SIGN UP

$20/mo

2GB Memory

2 Core Processor

40GB SSD Disk

3TB Transfer

SIGN UP

$40/mo

4GB Memory

2 Core Processor

60GB SSD Disk

4TB Transfer

SIGN UP

$80/mo

8GB Memory

4 Core Processor

80GB SSD Disk

5TB Transfer

SIGN UP

DigitalOcean

Log In

javajudd@gmail.com

•••••••

Log In

Forgot password?

Don't have an account? Sign Up

Thanks! Create your first Droplet now.

Looks like you don't have any Droplets.

Fortunately, it's very easy to create one.

Create Droplet

Droplets Images Networking API Support

Create Droplet

Applications & API

Your Tokens

Your Apps

Access

Personal Access Tokens

Generate new token

Tokens you have generated to access the DigitalOcean API

You have no tokens authorized to access your account.

Personal access tokens function like a combined name and password for API authentication.

Droplets Images Networking API Support

Create Droplet

New Personal Access Token

Back to Apps & API

Token Name

Enter Token Name
do-staging

Select Scopes

Read (Default) Write (Optional)

Generate Token

Scopes limit access for personal tokens. Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for DigitalOcean over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Droplets Images Networking API Support

Create Droplet

Applications & API

Your Tokens

Your Apps

Access

Personal Access Tokens

Generate new token

Tokens you have generated to access the DigitalOcean API

do-staging READ WRITE

77847e8e7d07c02135757096fdaaf8e9

Copy to Clipboard

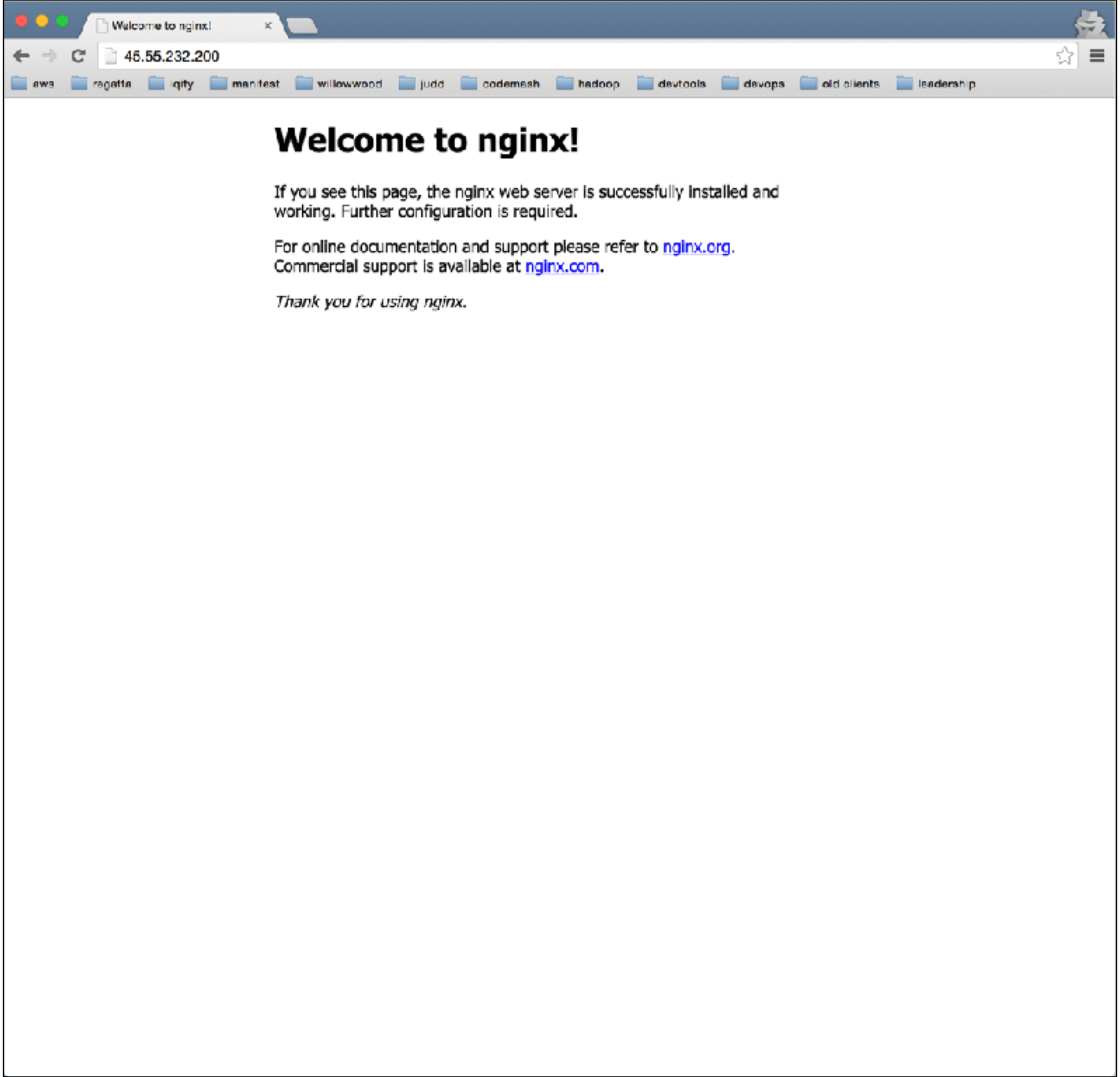
Personal access tokens function like a combined name and password for API authentication.

```
docker-machine create --driver digitalocean
   --digitalocean-access-token 77847e8e7d07c02135767096fdaaf8e do
```

```
Creating SSH key...
Creating Digital Ocean droplet...
To see how to connect Docker to this machine, run: docker-machine env do
```

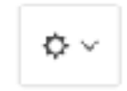
```
docker-machine env do
eval "$(docker-machine env do)”
docker run -d -p 80:80 nginx
docker-machine ip do
```

```
45.55.232.200
```


Welcome to nginx!
45.55.232.200
aws
regatta
qity
manifest
willowwood
judd
codemash
hadoop
devtools
devops
old clients
leadership
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.

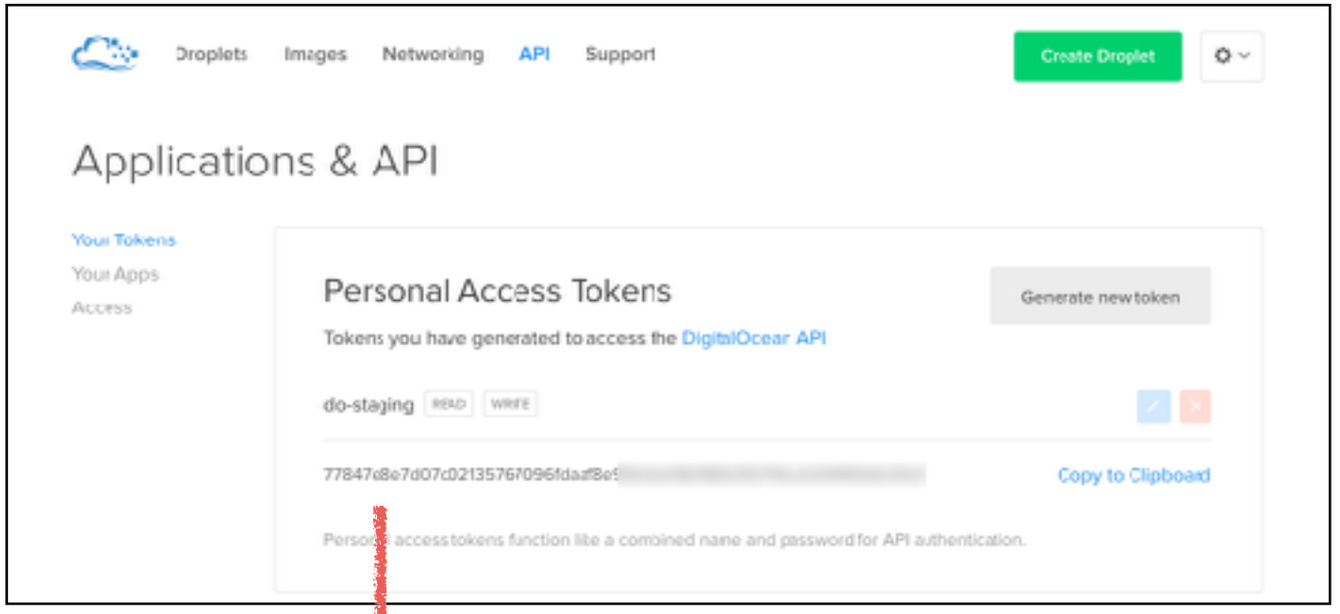
Droplets Images Networking API Support

Create Droplet

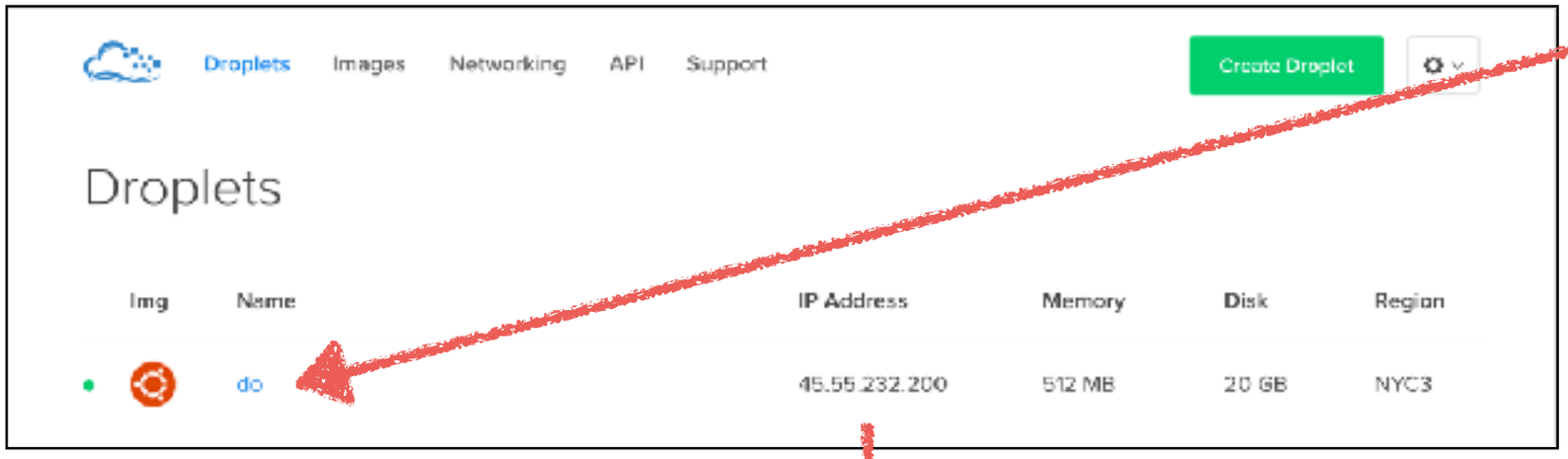


Droplets

	Img	Name	IP Address	Memory	Disk	Region
●		do	45.55.232.200	512 MB	20 GB	NYC3




```
docker-machine create --driver digitalocean
   --digitalocean-access-token 77847e8e7d07c02135767096fdaaf8e do
```



Lab DigitalOcean

1. Create a DigitalOcean account
2. Create a token
3. Create a DigitalOcean machine
4. Deploy a container
5. Remove the droplet


amazon
web services™

docker-machine

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Welcome to Identity and Access Management

IAM users sign-in link:

https://[redacted].signin.aws.amazon.com/console Customize | Copy Link

IAM Resources

Users: 1 | Roles: 5

Groups: 2 | Identity Providers: 0

Customer Managed Policies: 0

Security Status

3 out of 5 complete.

- Delete your root access keys
- Activate MFA on your root account
- Create individual IAM users
- Use groups to assign permissions
- Apply an IAM password policy

Feature Spotlight

Additional Information

IAM documentation

Web Identity Federation Playground

Policy Simulator

Videos, IAM release history and additional resources

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Create New Users | User Actions

Filter

Showing 1 results

	User Name	Groups	Password	Password Last Used	Access Keys	Creation Time
☐	[redacted]	1	✔	2015-10-23 08:59 EDT	None	2015-01-09 10..

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Summary

User ARN:	arn:aws:iam::[redacted]
Has Password:	Yes
Groups (for this user):	1
Path:	/
Creation Time:	2015-01-09 10:57 EST

Groups | **Permissions** | **Security Credentials**

This view shows all groups the User belongs to: **1 Group**

Add User to Groups

Group	Actions
developers	Remove from Group

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Summary

User ARN:	arn:aws:iam::[redacted]
Has Password:	Yes
Groups (for this user):	1
Path:	/
Creation Time:	2015-01-09 10:57 EST

Groups | **Permissions** | **Security Credentials**

Access Keys

Use access keys to make secure REST or Query protocol requests to any AWS service API. For your protection, you should never share your secret keys with anyone. In addition, industry best practice recommends frequent key rotation. Learn more about Access Keys

This user does not currently have any access keys.

Create Access Key

Sign-In Credentials

User Name	[redacted]
Password	Yes
Last Used	2015-10-23 08:59 EDT

Manage Password

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Summary

User ARN:	arn:aws:iam::[redacted]
Has Password:	Yes
Groups (for this user):	1
Path:	/
Creation Time:	2015-01-09 10:57 EST

Create Access Key

Your access key has been created successfully.

This is the last time these User security credentials will be available for download.

You can manage and recreate these credentials any time.

▶ Show User Security Credentials

Close | Download Credentials

ce API. For your protection,
tice recommends frequent

Sign-In Credentials

User Name	[redacted]
Password	Yes
Last Used	2015-10-23 08:59 EDT

Manage Password

Dashboard

Details

Groups

Users

Roles

Policies

Identity Providers

Account Settings

Credential Report

Encryption Keys

Summary

User ARN:	arn:aws:iam::[redacted]
Has Password:	Yes
Groups (for this user):	1
Path:	/
Creation Time:	2015-01-09 10:57 EST

ce API. For your protection,
tice recommends frequent

Last Used 2015-10-23 08:59 EDT

Create Access Key

Your access key has been created successfully.

This is the last time these User security credentials will be available for download.

You can manage and recreate these credentials any time.

▼ Hide User Security Credentials

[redacted]

Access Key ID: AKIAI[redacted]

Secret Access Key: Dbig4+636[redacted]

Close | Download Credentials

VPC Dashboard

Filter by VPC:

None

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Peering Connections

Security

Network ACLs

Security Groups

VPN Connections

Customer Gateways

Virtual Private Gateways

VPN Connections

Create VPC | Actions

Search VPCs and their prope

1 to 1 of 1 VPC

Name	VPC ID	State	VPC CIDR	DHCP options set	Route table
vpc-public	vpc-c602eaa2	available	10.0.0.0/16	dopt-70829f12	rtb-646fbc00

vpc-c602eaa2 (10.0.0.0/16) | vpc-public

Summary | Flow Logs | Tags

VPC ID:	vpc-c602eaa2 | vpc-public	**Network ACL:**	acl-dadd29be
State:	available	**Tenancy:**	Default
VPC CIDR:	10.0.0.0/16	**DNS resolution:**	yes
DHCP options set:	dopt-70829f12	**DNS hostnames:**	yes

```
docker-machine -D create \
    --driver amazonec2 \
    --amazonec2-access-key AKIAIMN \
    --amazonec2-secret-key Dbig4+6364 \
    --amazonec2-region us-east-1 \
    --amazonec2-vpc-id vpc-c602eaa2 \
    --amazonec2-zone b \
    aws
```

```
docker-machine env aws
eval "$(docker-machine env aws)”
docker run -d -p 80:80 nginx
docker-machine ip aws
```

52.23.222.169

EC2 Dashboard
Events
Tags
Reports
Limits

INSTANCES
Instances
Spot Requests
Reserved Instances
Commands

IMAGES
AMIs
Bundle Tasks

ELASTIC BLOCK STORE
Volumes
Snapshots

NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces

LOAD BALANCING
Load Balancers

AUTO SCALING
Launch Configurations

Launch Instance | Connect | Actions

Filter by tags and attributes or search by keyword

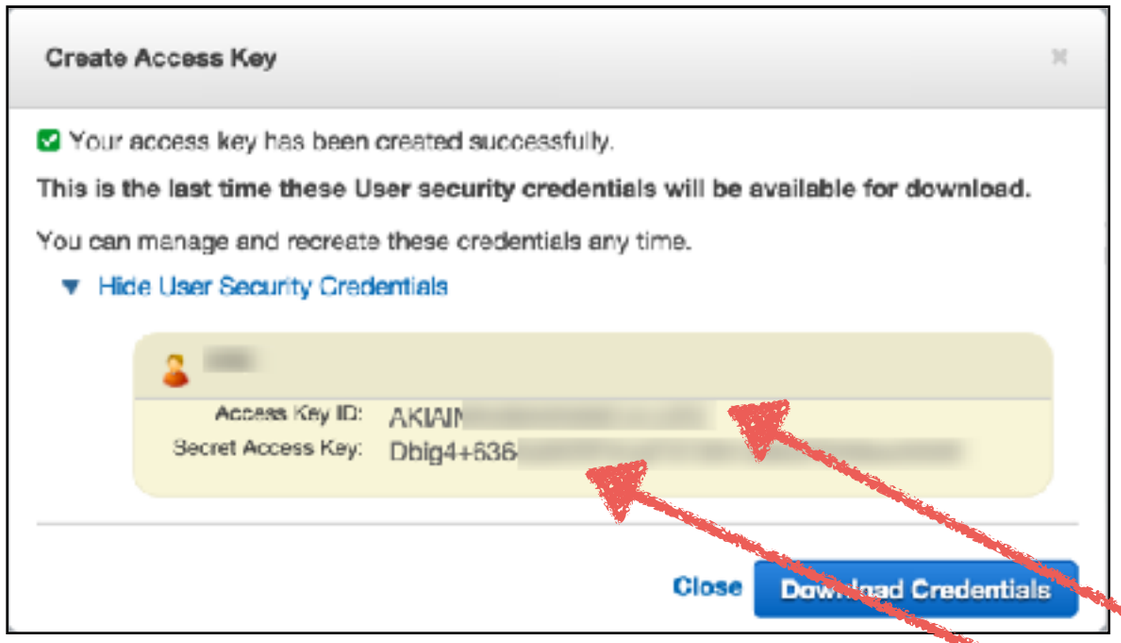
1 to 1 of 1

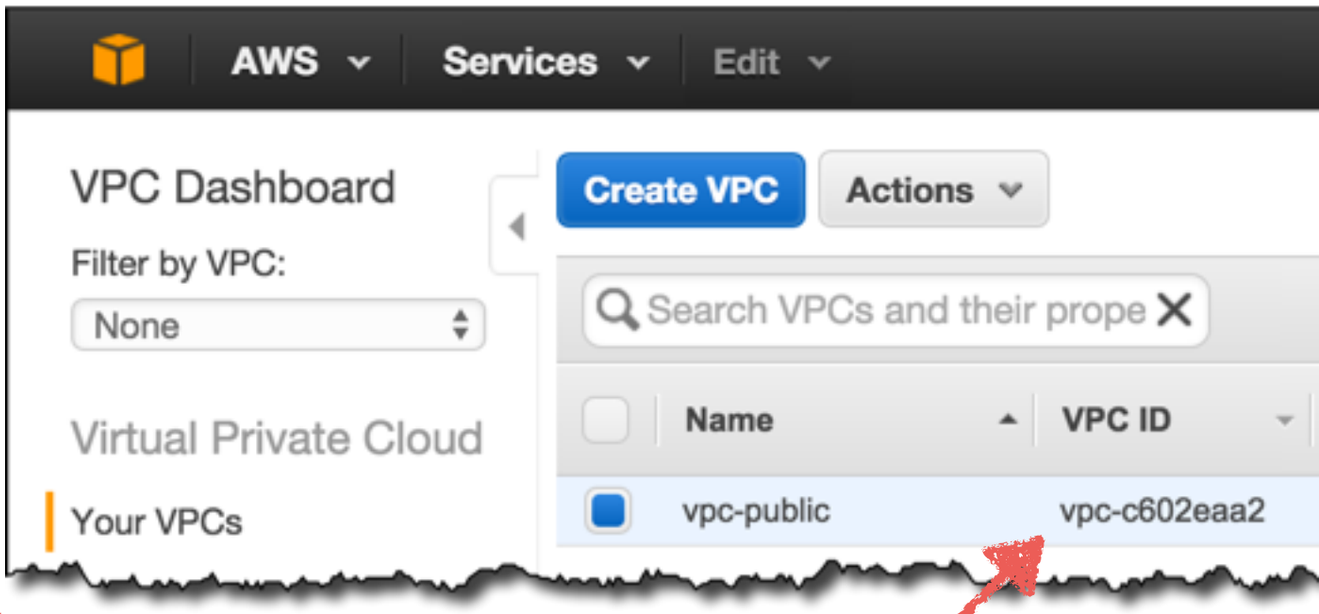
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
aws	i-0707a0b7	t2.micro	us-east-1b	running	2/2 checks ...

Instance: i-0707a0b7 (aws) **Public DNS:** ec2-52-23-222-169.compute-1.amazonaws.com

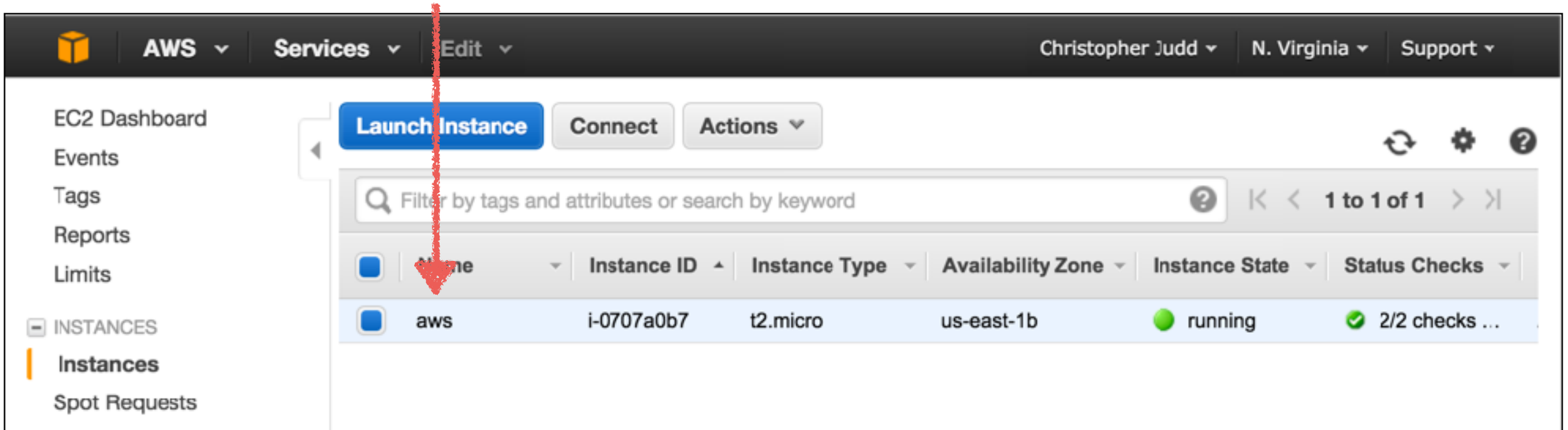
Description | Status Checks | Monitoring | Tags

Instance ID	i-0707a0b7	**Public DNS**	ec2-52-23-222-169.compute-1.amazonaws.com
Instance state	running	**Public IP**	52.23.222.169
Instance type	t2.micro	**Elastic IP**	-
Private DNS	ip-10-0-0-	**Availability zone**	us-east-1b






```
docker-machine -D create \
    --driver amazonec2 \
    --amazonec2-access-key AKIAIMN \
    --amazonec2-secret-key Dbig4+6364 \
    --amazonec2-region us-east-1
    --amazonec2-vpc-id vpc-c602eaa2 \
    --amazonec2-zone b \
    aws
```



EC2 Dashboard
Events
Tags
Reports
Limits

INSTANCES
Instances
Spot Requests
Reserved Instances
Commands

IMAGES
AMIs
Bundle Tasks

ELASTIC BLOCK STORE
Volumes
Snapshots

NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces

LOAD BALANCING
Load Balancers

AUTO SCALING
Launch Configurations

Create Security Group | Actions

search : sg-d477e2b2 | Add filter | 1 to 1 of 1

Name	Group ID	Group Name	VPC ID	D
	sg-d477e2b2	docker-machine	vpc-c602eaa2	D

Security Group: sg-d477e2b2

Description | **Inbound** | Outbound | Tags

Edit

Type	Protocol	Port Range	Source
SSH	TCP	22	0.0.0.0/0
Custom TCP Rule	TCP	2376	0.0.0.0/0

EC2 Dashboard
Events
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
Commands
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
AUTO SCALING
Launch Configurations

Create Security Group | Actions

search : sg-d477e2b2 | Add filter | 1 to 1 of 1

Name	Group ID	Group Name	VPC ID
	sg-d477e2b2	docker-machine	vpc-c602eaa2

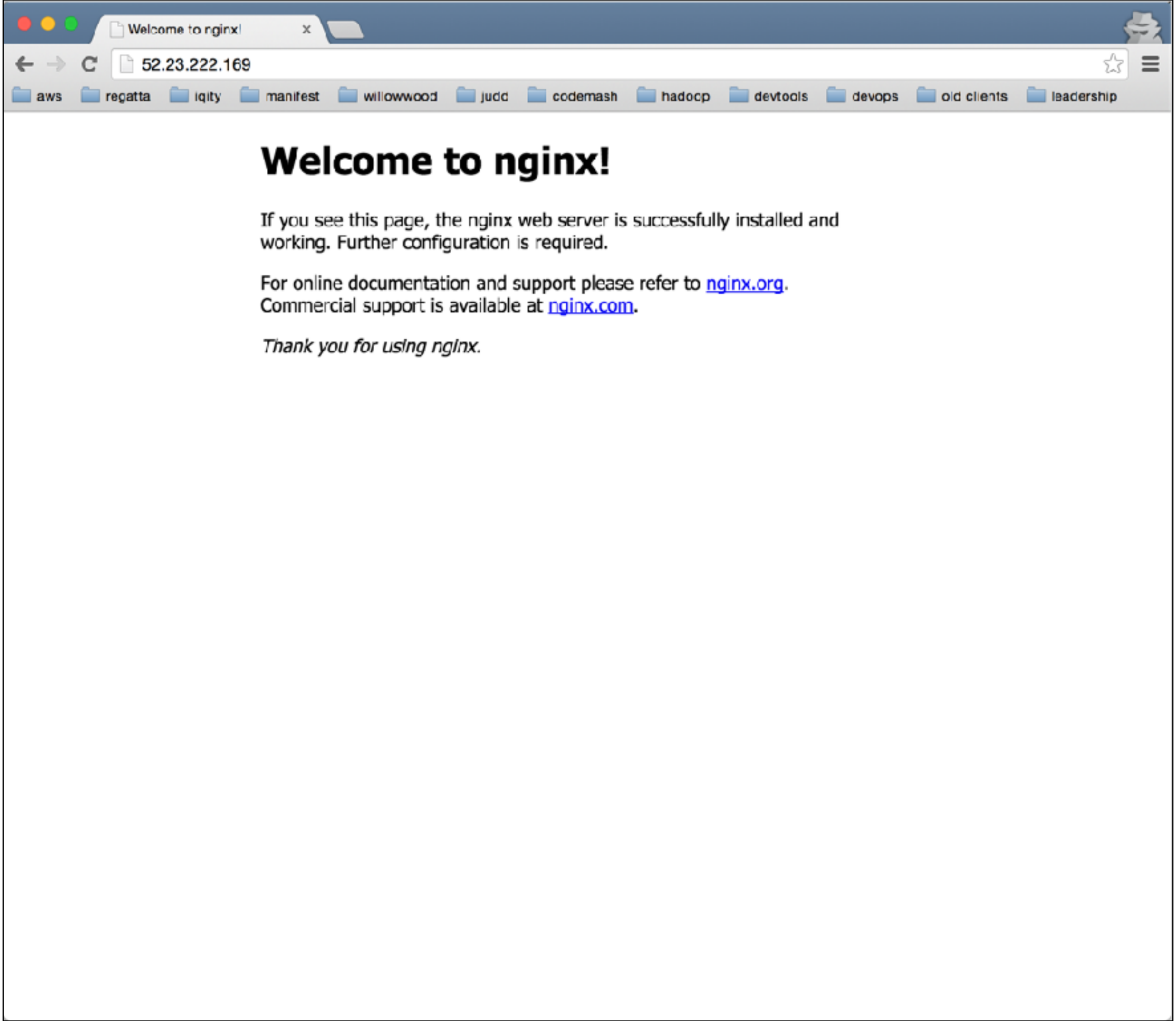
Edit inbound rules

Type	Protocol	Port Range	Source	
SSH	TCP	22	Anywhere	0.0.0.0/0
Custom TCP Rule	TCP	2376	Anywhere	0.0.0.0/0
HTTP	TCP	80	Anywhere	0.0.0.0/0

Add Rule

Cancel | Save

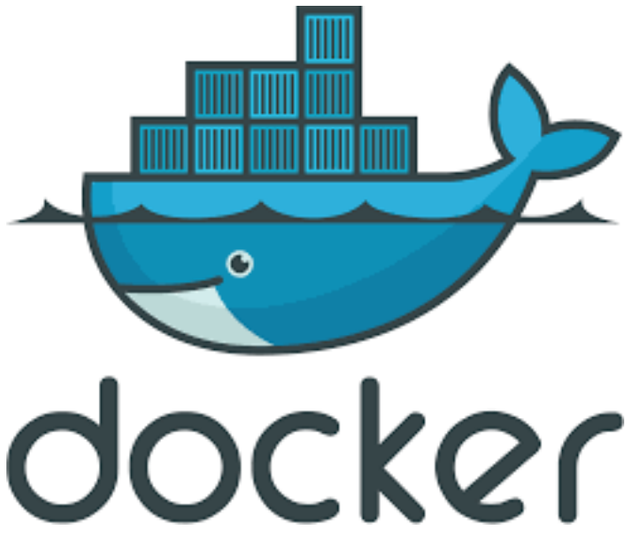
SSH	TCP	22	0.0.0.0/0
Custom TCP Rule	TCP	2376	0.0.0.0/0

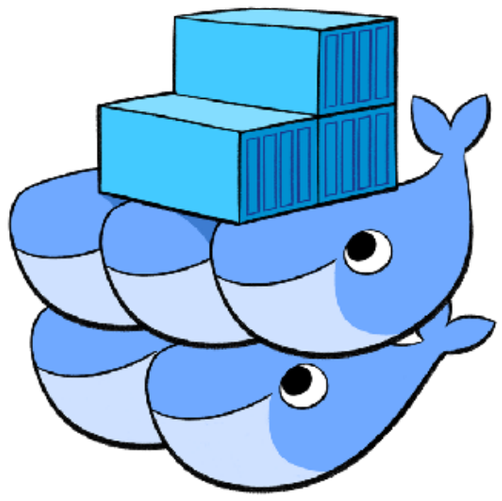

Welcome to nginx!
52.23.222.169
aws
regatta
iqity
manifest
willowwood
judd
codemash
hadocp
devtools
devops
old clients
leadership
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.

Lab AWS

1. Create an AWS account
2. Create an access key
3. Find VPC ID
4. Create AWS machine
5. Add port to Security Group
6. Terminate EC2 instance

Summary



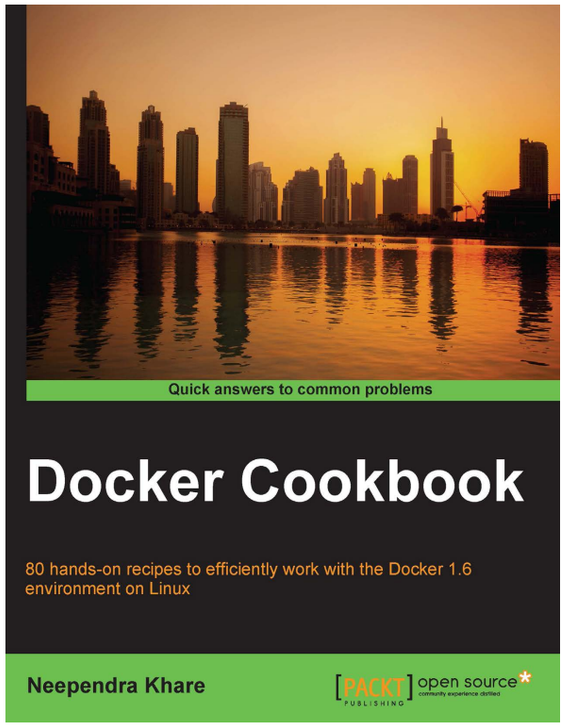


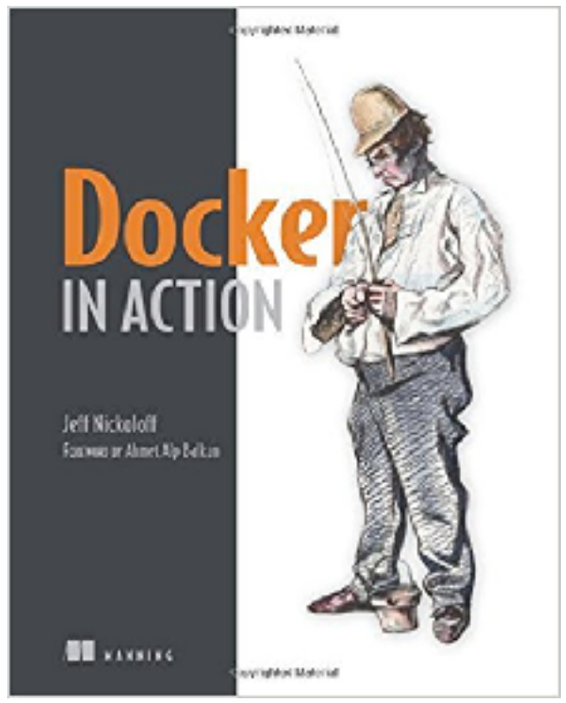
https://github.com/coreos/rkt

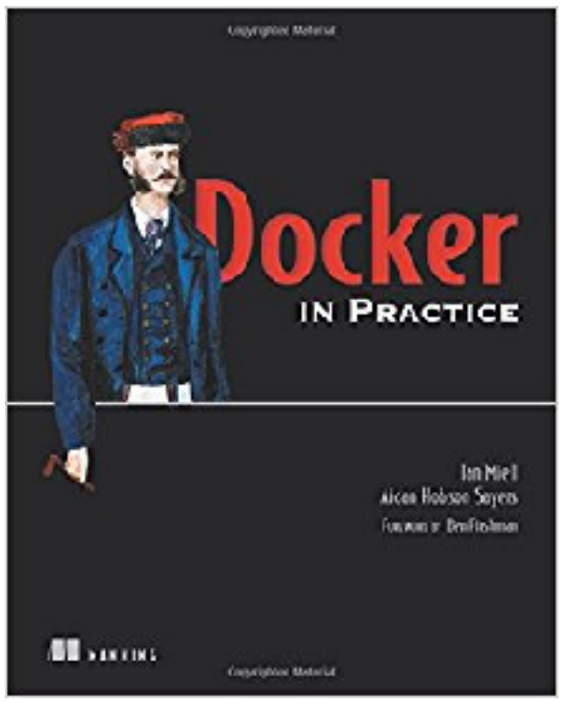
https://www.flickr.com/photos/glynlowe/10921732025/

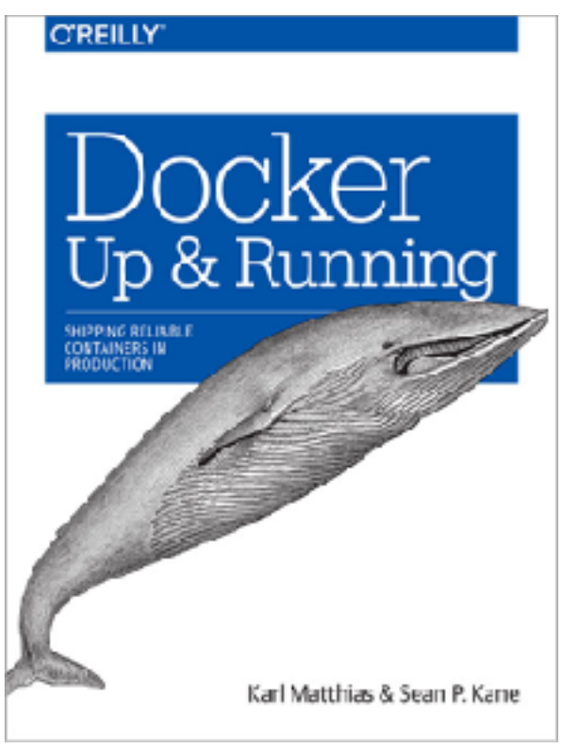

AVERY BREWING CO
Welcome
AVERY BREWING CO
MEET THE BREWERY:
Odell Brewing Co.

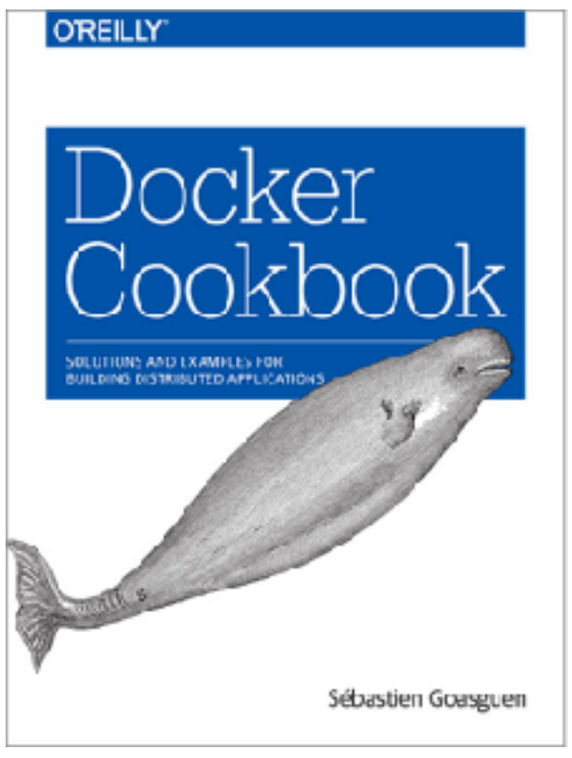
Resources


Quick answers to common problems
Docker Cookbook
80 hands-on recipes to efficiently work with the Docker 1.6 environment on Linux
Neependra Khare
[PACKT] open source
PUBLISHING


Docker
IN ACTION
Jeff Nickoloff
MANNING


Docker
IN PRACTICE
MANNING


O'REILLY
Docker
Up & Running
SHIPPING RELIABLE CONTAINERS IN PRODUCTION
Karl Matthias & Sean P. Kane


O'REILLY
Docker
Cookbook
SOLUTIONS AND EXAMPLES FOR BUILDING DISTRIBUTED APPLICATIONS
Sébastien Goasguen

O'REILLY®

Docker for Java Developers

Package, Deploy, and Scale with Ease

Arun Gupta

O'REILLY®

Compliments of NGINX

Kubernetes for Java Developers

Orchestrate Multi-Container Applications with Ease

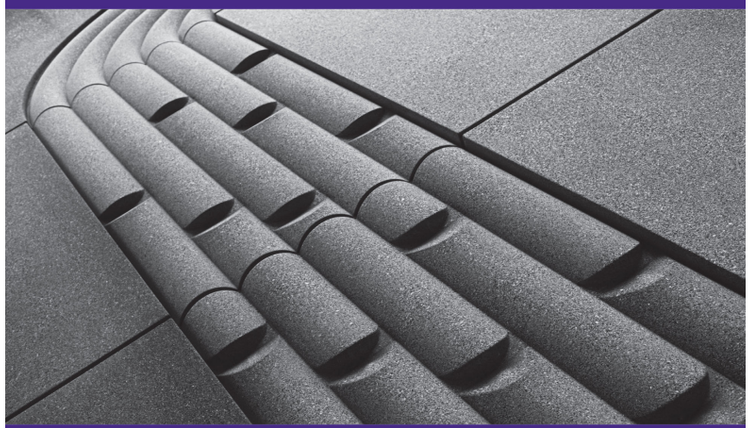
Arun Gupta

O'REILLY®

Compliments of NGINX

Containerizing Continuous Delivery in Java

Docker Integration for Build Pipelines and Application Architecture



Daniel Bryant

Getting Started With Docker

BY CHRISTOPHER JUDD

CONTENTS

- About Docker
- Docker Architecture
- Other Helpful Commands
- Docker Machine
- Enterprise Docker

ABOUT DOCKER

Almost overnight, Docker has become the de facto standard that developers and system administrators use for packaging, deploying, and running distributed and cloud native applications. It provides tools for simplifying DevOps by enabling developers to create templates called images that can be used to create lightweight virtual machines called containers, which include their applications and all of their applications' dependencies. These lightweight virtual machines can be promoted through testing and production environments where sysadmins deploy and run them.

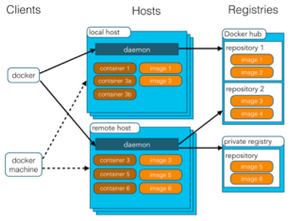
Docker makes it easier for organizations to automate infrastructure, isolate applications, maintain consistency, and improve resource utilizations.

Like the popular version control software Git, Docker has a social aspect in that developers and sysadmins can share their images via Docker Hub.

Docker is an open source solution that is available as the free Community Edition or the subscription based Enterprise Edition for multiple platforms. Docker runs natively on Linux since Docker was originally built on Linux containers but it also works on Mac and Windows. Many enterprise tools have also grown up around Docker to make it easier to manage and orchestrate complex distributed and clustered applications architectures.

DOCKER ARCHITECTURE

Docker utilizes a client-server architecture and a remote API to manage and create Docker containers and images. Docker containers are created from Docker images. The relationship between containers and images are analogous to the relationship between objects and classes in object-oriented programming, where the image describes the container and the container is a running instance of the image.



Docker Images	A recipe or template for creating Docker containers. It includes the steps for installing and running the necessary software
Docker Container	Like a tiny virtual machine that is created from the instructions found within the Docker image
Docker Client	Command-line utility or other tool that takes advantage of the Docker API (docs.docker.com/reference/api/docker_remote_api) to communicate with a Docker daemon
Docker Host	A physical or virtual machine that is running a Docker daemon and contains cached images as well as runnable containers created from images
Docker Registry	A repository of Docker images that can be used to create Docker containers. Docker Hub (hub.docker.com) is the most popular social example of a Docker repository.
Docker Machine	A utility for managing multiple Docker hosts, which can run locally in VirtualBox or remotely in a cloud hosting service such as Amazon Web Services, Microsoft Azure, Google Cloud Platform, or Digital Ocean.

Getting Started With

Kubernetes

UPDATED BY CHRIS JUDD
ORIGINAL BY ARUN GUPTA

CONTENTS

- What Is Kubernetes?
- Key Concepts of Kubernetes
- Kubernetes Architecture
- Getting Started With Kubernetes
- Run Your First Container
- Scale Applications

WHAT IS KUBERNETES?

Kubernetes (kubernetes.io) is an open source orchestration system for managing containerized applications across multiple hosts, providing basic mechanisms for the deployment, maintenance, and scaling of applications. Originally created by Google, in March of 2016 it was donated to the Cloud Native Computing Foundation (CNCF).

Kubernetes, or "k8s" or "kube" for short, allows the user to declaratively specify the desired state of a cluster using high-level primitives. For example, the user may specify that they want three instances of the Couchbase server container running. Kubernetes' self-healing mechanisms, such as auto-restarting, re-scheduling, and replicating containers then converge the actual state towards the desired state.

Kubernetes supports Docker and Rocket containers. An abstraction around the containerization layer will allow for other container image formats and runtimes to be supported in the future.

KEY CONCEPTS OF KUBERNETES

POD

A Pod is the smallest deployable unit that can be created, scheduled, and managed. It's a logical collection of containers that belong to an application.

Each resource in Kubernetes is defined using a configuration file. For example, a Couchbase pod can be defined with the following `.yaml` file:

```
apiVersion: v1
kind: Pod
# labels attached to this Pod
metadata:
  name: couchbase-pod
  labels:
    name: couchbase-pod
spec:
  containers:
  - name: couchbase
    # Docker image that will run in this Pod
    image: couchbase
    ports:
    - containerPort: 8091
```

LABEL

A label is a key/value pair that is attached to objects, such as pods. In the previous example, `metadata.labels` define the labels attached to the pod.

Labels define identifying attributes for the object and is only meaningful and relevant to the user. Multiple labels can be attached to a resource. Labels can be used to organize and to select subsets of objects.

REPLICA SETS

A replica set ensures that a specified number of pod replicas are running on worker nodes at any one time. It allows both up- and down-scaling the number of replicas. It also ensures recreation of a pod when the worker node reboots or otherwise fails.

NOTE: Replica Sets replaces Replication Controllers.

A Replica Set creating two instances of a Couchbase pod can be defined as:

```
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: couchbase-rs
spec:
  # Two replicas of the Pod to be created
  replicas: 2
  # Identifies the label key and value on the Pod that
  # this Replica Set is responsible for managing
  selector:
    matchLabels:
      app: couchbase-rs-pod
    matchExpressions:
      - {key: tier, operator: In, values: ["backend"]}
template:
  metadata:
    labels:
      # label key and value on the pod.
      # These must match the selector above.
      app: couchbase-rs-pod
      tier: backend
  spec:
    containers:
    - name: couchbase
      image: couchbase
      ports:
      - containerPort: 8091
```


CONTENTS

- What is a Linux Container?
- Docker's Architecture
- How is Docker different than JVM?
- Docker's Benefits
- Docker Basic Workflow... and more!

Java Containerization

BY AMJAD AFANAH

WHAT IS A LINUX CONTAINER?

A Linux container is an operating system level virtualization technology that, unlike a virtual machine (VM), shares the host operating system kernel and makes use of the guest operating system system libraries for providing the required OS capabilities. Since there is no dedicated operating system, containers are more lightweight and start much faster than VMs.

In 2013, Docker was developed as an open platform for packaging, deploying, and running distributed applications. Docker uses its own Linux container library called libcontainer. It has become the most popular and widely used container management system.

This Refcard will focus on the design, deployment, service discovery and management of Java applications on Docker.

DOCKER'S ARCHITECTURE

Docker uses a typical client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon communicate via sockets or through a RESTful API.

REFERENCE	DESCRIPTION
Docker Images	Read-only templates that use union file systems to combine layers—making them very lightweight. Images are built from Dockerfiles.
Docker Registries	Store Docker images. Users can push (or publish) their images to a public registry (like Docker Hub) or to their own registry behind a firewall. Registries store the "tagged" images—allowing users to maintain different versions of the same image.
Docker Containers	Virtualized application environments that run on a Docker Host in isolation. Containers are launched from Docker images, adding a read-write layer on top of the image (using a union file system) as well as the network/bridge interface and IP address. When a container is launched, the process specified in the Dockerfile is executed and the logs are captured for auditing and diagnostics.
Dockerfiles	Composed of various commands (instructions) listed successively to automatically perform actions on a base image in order to create a new one. The instructions specify the operating system, application artifacts, data volumes, and exposed ports to be used, as well as the command (or script) to run when launching a Docker container.
Docker Host	A Linux host (either a physical/bare-metal server or a virtual machine) that is running a Docker daemon on which images can be built, pulled or pushed and containers can run in isolation.
Docker Client	Command-line utility or other tool that takes advantage of the Docker API (docs.docker.com/reference/api/docker_remote_api) to communicate with a Docker daemon

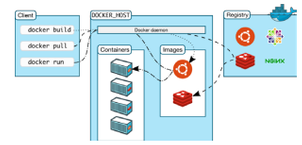


FIGURE 1: DOCKER ARCHITECTURE

HOW IS DOCKER DIFFERENT THAN JVM?

The JVM is Java's solution for application portability across different platforms—but Docker provides a different kind of virtualization that makes use of the guest operating system system libraries, and not just the Java application. When a Docker container is launched, a filesystem is allocated, along with the network/bridge interface and IP address. The command (or script) specified in the Dockerfile used to build the underlying Docker image is then executed and the resulting Linux process then runs in isolation.

As a result, Docker can be used to package an entire JVM along with the JAR or WAR files and other parts of the application into a single container that can run on any Linux host consistently. This eliminates some of the challenges associated with making sure that the right JAR file version is used on the right JVM. Moreover, CPU and Memory resource controls can be used with Docker containers—allowing users to allocate maximum amounts of resources to allocate for an application.

DOCKER'S BENEFITS

The main advantages of Docker are:

Application Portability—Docker containers run exactly the same on any Linux host. This eliminates the challenge of deploying

JAVA CONTAINERIZATION

Personal Open source Business Explore Pricing Blog Support This repository Search Sign in Sign up

docker / labs

Watch 104 Star 1,066 Fork 193

Code Issues 1 Pull requests 0 Projects 0 Pulse Graphs

This is a collection of tutorials for learning how to use Docker with various tools. Contributions welcome.

168 commits 4 branches 0 releases 25 contributors

Branch: master New pull request Find file Clone or download

ManoMarks committed on GitHub Merge pull request #91 from arun-gupta/nginx ... Latest commit 2eea1fc 2 days ago

12factor	Fix broken URL (#61)	27 days ago
Docker-Orchestration	Update readme.md	6 months ago
beginner	Updating to reflect changes in the Example Voting App	8 days ago
developer-tools	Update README.md	9 days ago
dockercon-us	update "swarm computing" to "swarm mode"	4 months ago
security	Security hands-on lab from dockercon (#71)	17 days ago
slides	updating slides for NGINX webinar	2 days ago
swarm-mode	Pass the correct listen address to workers. (#85)	9 days ago
windows	Update Setup-Win10.md	8 days ago
.gitignore	First version of Node JS debugging in Visual Studio Code tutorial	16 days ago
.gitmodules	removing java-debuggin submodule	2 months ago
LICENSE	adding basic labs files	6 months ago
README.md	Fixing a few broken links	15 days ago
contribute.md	adding trademark guidelines to contribute.md	6 months ago

README.md

Docker Tutorials and Labs

This repo contains Docker labs and tutorials authored both by Docker, and by members of the community. We welcome contributions and want to grow the repo.

https://github.com/docker/labs

Play with Docker classroom

About

beginner intermediate advanced

others

Beginner

- Hello World (Developer Operations)
- First Alpine Linux Container (Linux Operations Developer)
- Simple Web App (Linux Developer)
- Swarm stack introduction (Linux Operations)
- Docker images deeper dive (Developer Operations Linux)
- Docker containers deeper dive (Linux Developer Operations)
- Docker Volumes (Linux Developer Operations)
- Swarm mode introduction (Linux Operations)
- Docker compose with swarm secrets (Developer Operations Linux)
- Docker swarm config files (Developer Operations Linux)
- Windows Containers Setup (Windows Operations Developer)
- Windows Containers Basics (Windows Operations Developer)
- Docker Orchstration Hands-on Lab (Operations Networking)
- Windows Containers Multi-Container Applications

(Windows Operations Developer)

Intermediate

- Service Discovery under Docker Swarm Mode (Linux Operations)
- Docker volume sshfs (Linux Operations Plugins Volumes)

Learn more →

Want to take an in-depth, official Docker training course? Check out training.docker.com

Register for DockerCon! - http://europe.dockercon.com/

docker

Join the docker community on Slack! Connect with your peers, share ideas and ask questions - Register here

http://training.play-with-docker.com/alacart/

Christopher M. Judd

Manifest Solutions

CTO and Partner

email: javajudd@gmail.com

web: www.juddsolutions.com

blog: juddsolutions.blogspot.com

twitter: javajudd