



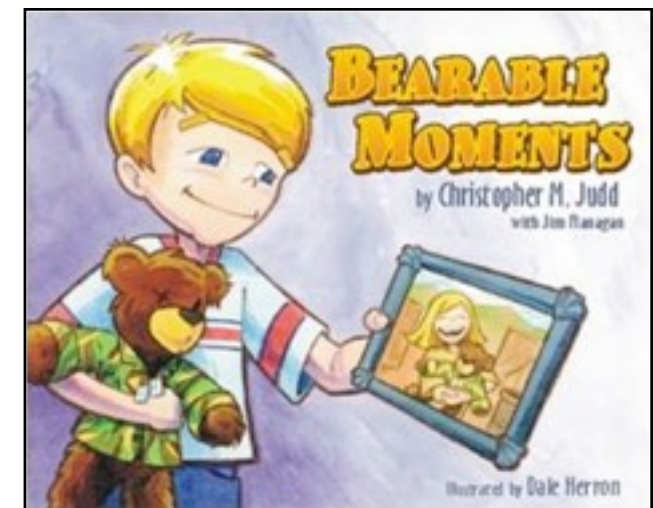
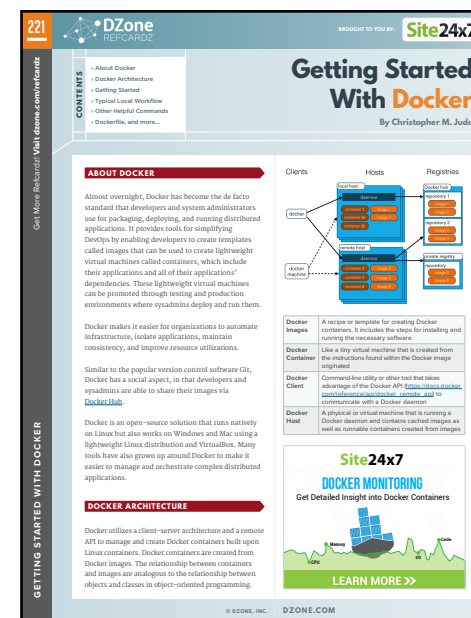
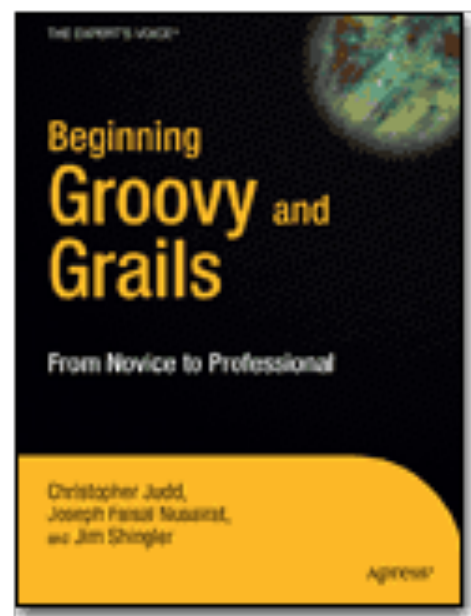
Christopher M. Judd

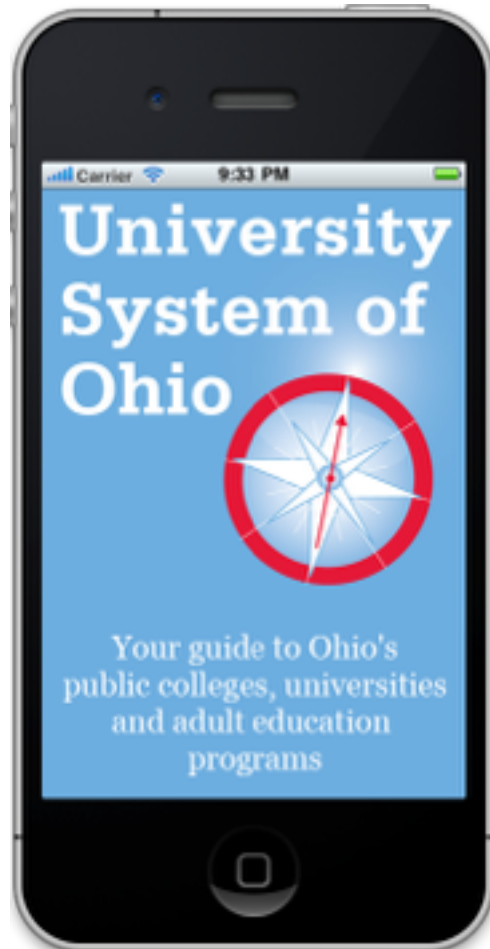
Christopher M. Judd

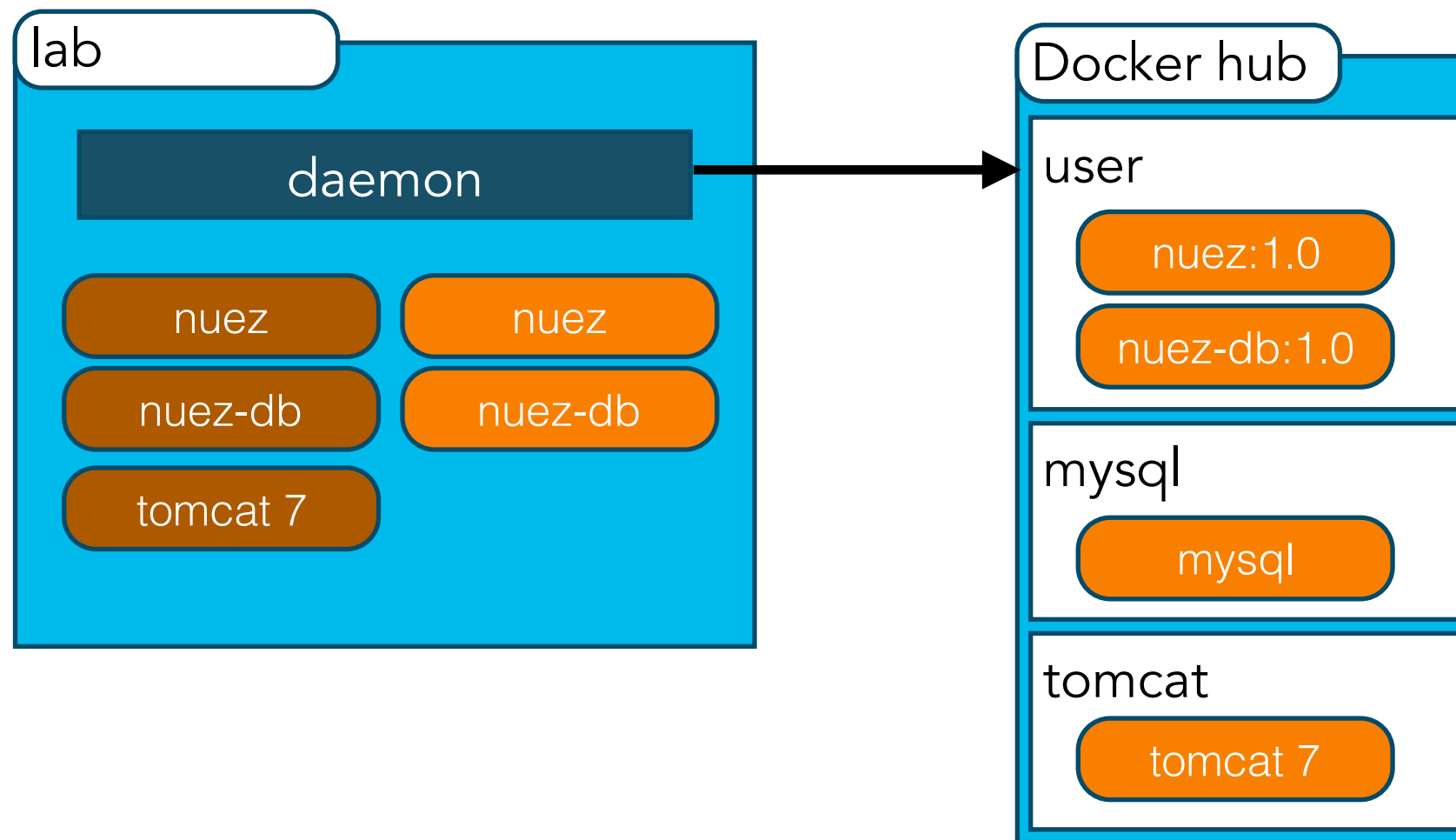
CTO and Partner at

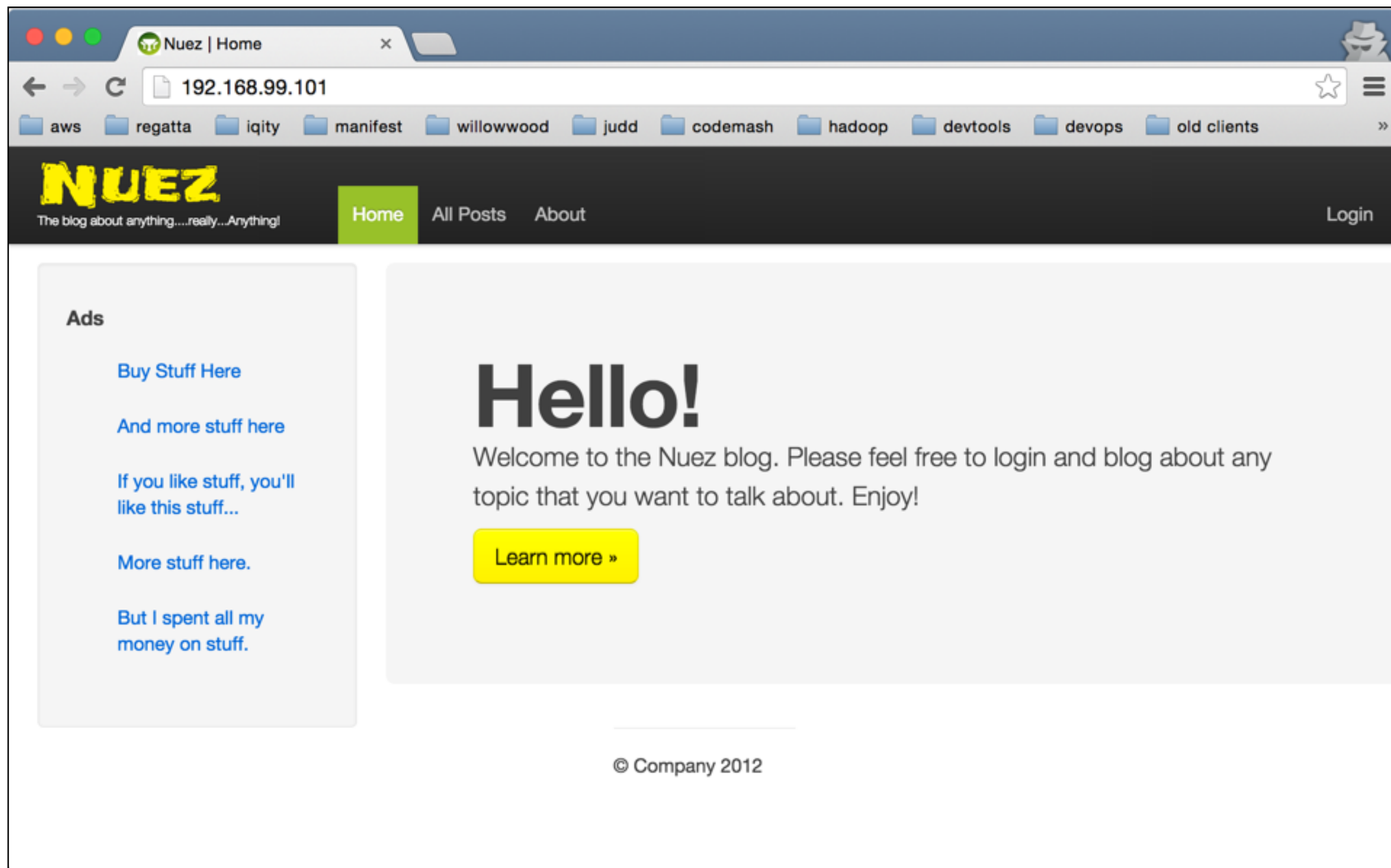


Central Ohio Java Users Group leader











[Pull requests](#) [Issues](#) [Gist](#)



[Contributions](#) [Repositories](#) [Public activity](#)

[Edit profile](#)

[Search](#)

[All](#) [Public](#) [Private](#) [Sources](#) [Forks](#) [Mirrors](#) [New](#)

nuez-compose

Docker composition for the [nuez-db](#) and [nuez](#) web application.

Updated 44 minutes ago

nuez-docker

An example of a docker file that creates an image for [nuez](#) from a URL.

Updated 20 hours ago

nuez

Y forked from [zendern/nuez](#)

Updated 21 hours ago

nuez-db

Docker configuration for [nuez-db](#)

Updated a day ago

Christopher M. Judd
cjudd

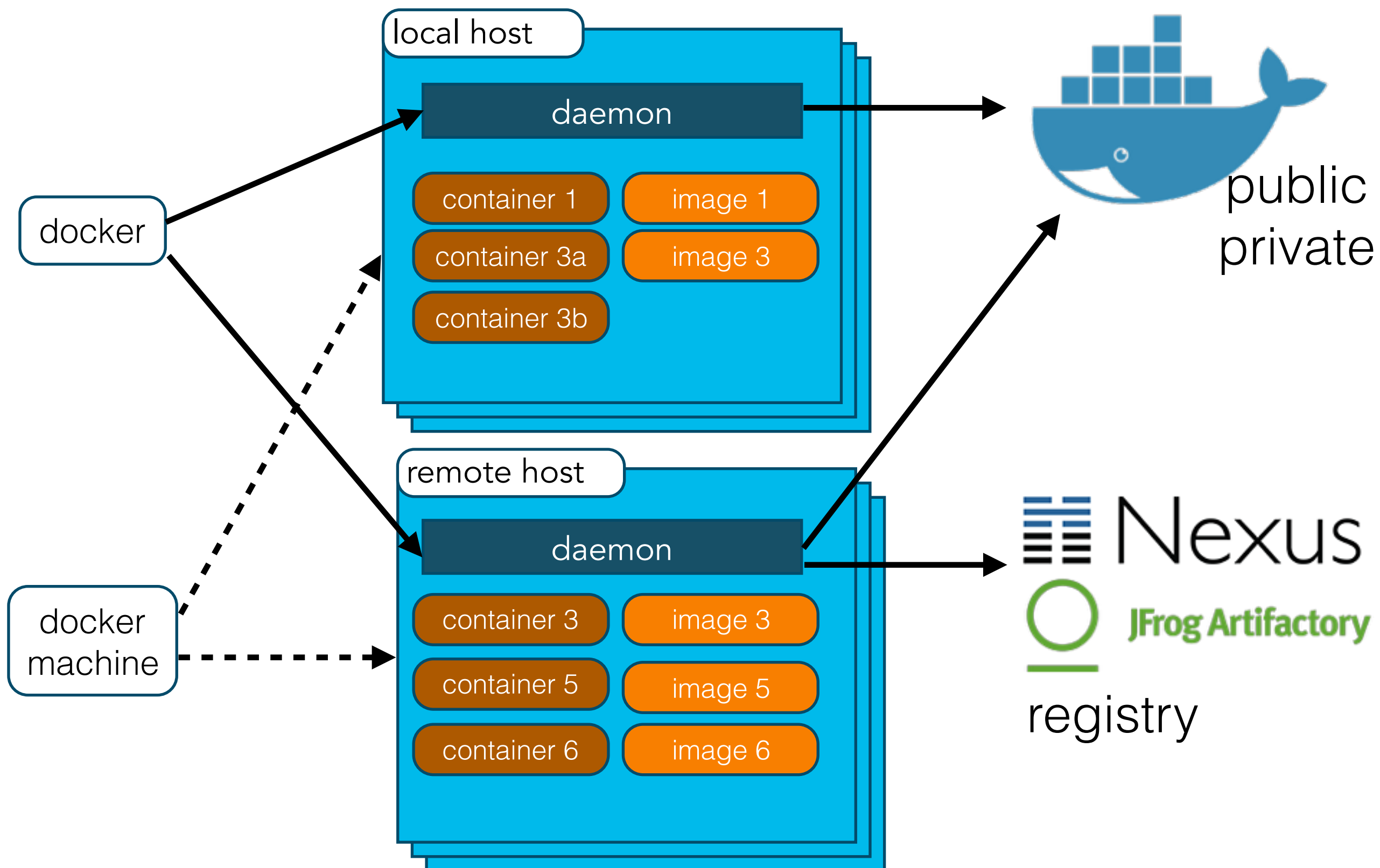
 Judd Solutions
 Columbus Ohio
 github@juddsolutions.com
 <http://juddsolutions.blogspot.c...>
 Joined on Jan 3, 2009

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

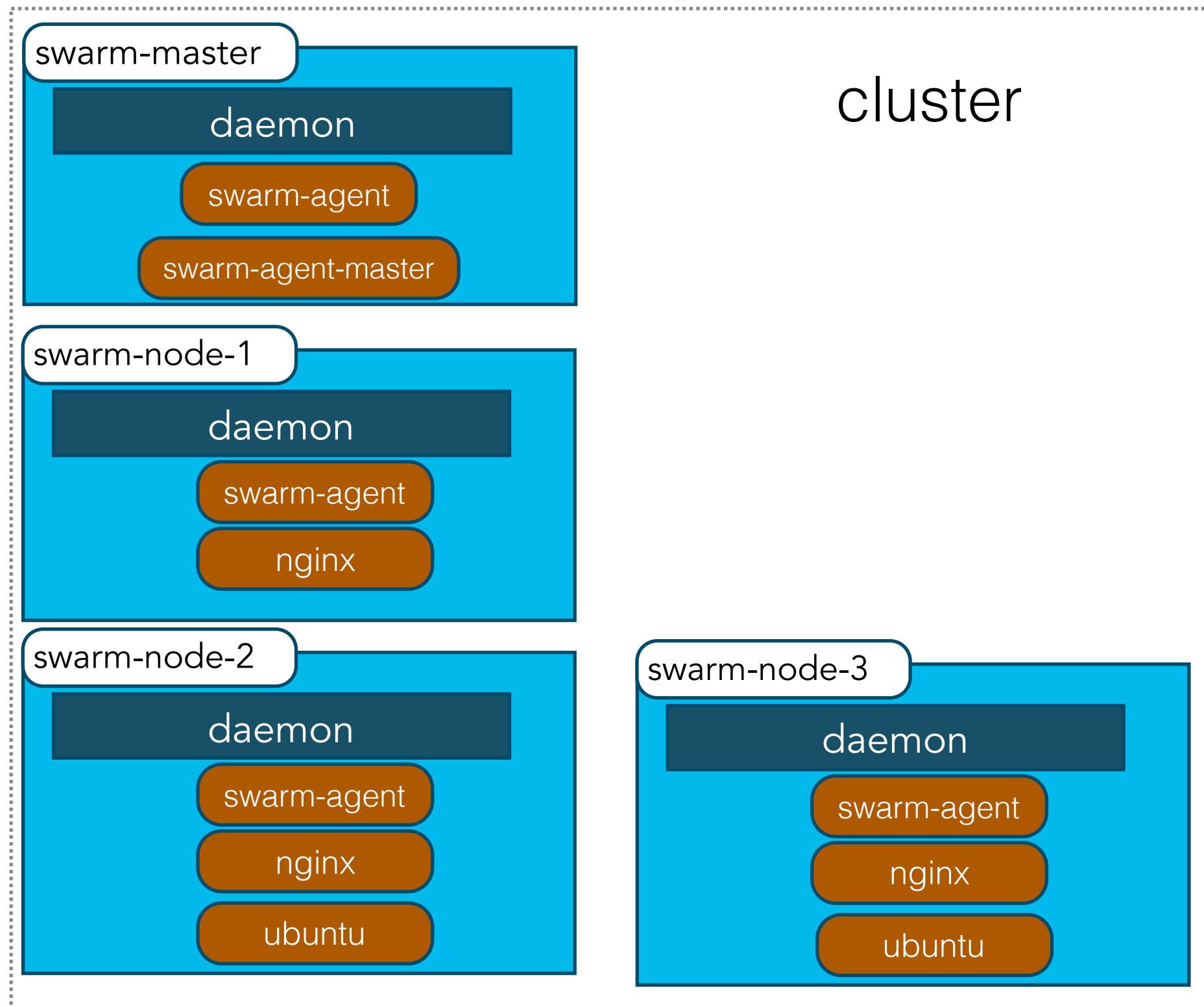
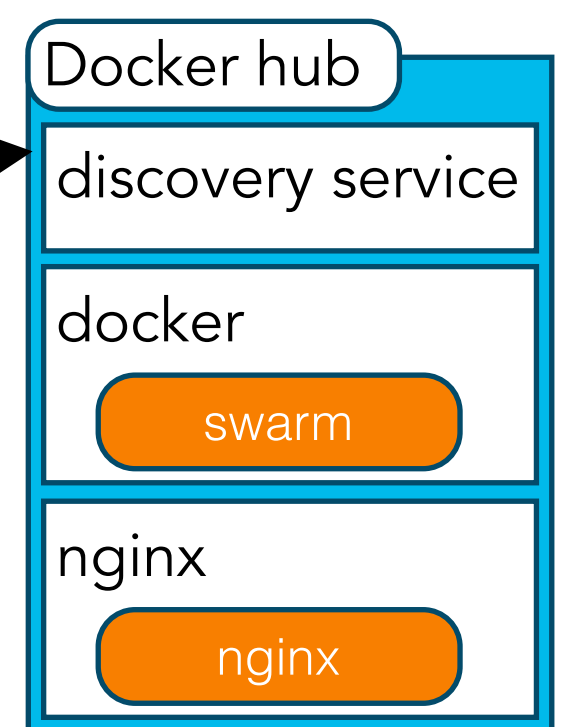
Organizations

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



Composing Docker API





DigitalOcean



amazon
web services™

Docker for Devs Workshop

<https://s3.amazonaws.com/cmj-presentations/docker-codemash-2016/index.html>

aws regatta iqity manifest willowwood judd codemash hadoop devtools devops



docker

for Devs

by Christopher Judd

[Slides](#)
[Commands](#)
[Docker DZone Refcardz](#)
[Prerequisites](#)

- [Docker Toolbox \(Mac or Windows\)](#)
- [Docker Install \(Linux\)](#)
- [Docker Hub Account](#)

[Code](#)

- [nuez \(web app\)](#)
- [nuez-db \(mysql database image\)](#)
- [nuez-docker \(nuez web application image\)](#)
- [nuez-compose \(docker composer for nuez & nuez-db\)](#)

<https://s3.amazonaws.com/cmj-presentations/docker-codemash-2016/index.html>

CONTENTS

- » About Docker
- » Docker Architecture
- » Getting Started
- » Typical Local Workflow
- » Other Helpful Commands
- » Dockerfile, and more...

Getting Started With Docker

By Christopher M. Judd

ABOUT DOCKER

Almost overnight, Docker has become the de facto standard that developers and system administrators use for packaging, deploying, and running distributed 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.

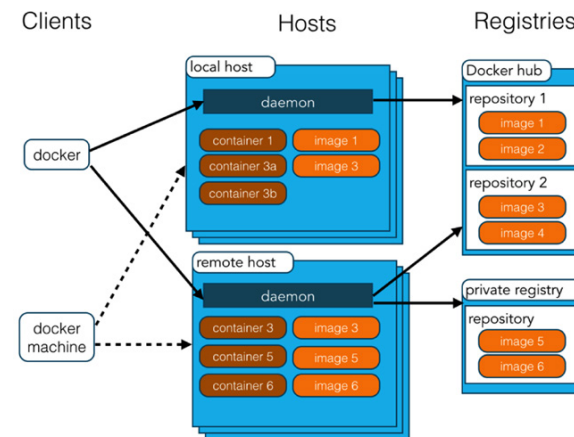
Docker makes it easier for organizations to automate infrastructure, isolate applications, maintain consistency, and improve resource utilizations.

Similar to the popular version control software Git, Docker has a social aspect, in that developers and sysadmins are able to share their images via [Docker Hub](https://hub.docker.com/).

Docker is an open-source solution that runs natively on Linux but also works on Windows and Mac using a lightweight Linux distribution and VirtualBox. Many tools have also grown up around Docker to make it easier to manage and orchestrate complex distributed applications.

DOCKER ARCHITECTURE

Docker utilizes a client-server architecture and a remote API to manage and create Docker containers built upon Linux containers. 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.

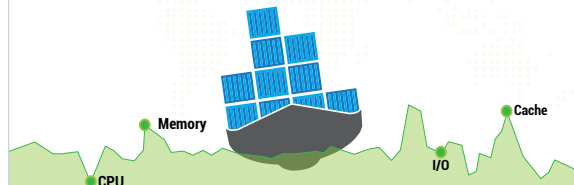


Docker Images	A recipe or template for creating Docker containers. It includes the steps for installing and running the necessary software.
Docker Container	Like a tiny virtual machine that is created from the instructions found within the Docker image originated
Docker Client	Command-line utility or other tool that takes advantage of the Docker API (https://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 MONITORING

Get Detailed Insight into Docker Containers



LEARN MORE >>

MY JOURNEY TO ENLIGHTENMENT



FREEK BOX

FREEK BOX

Start

Ivan 



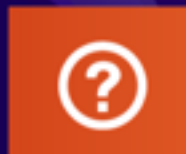
Skype



Reading List



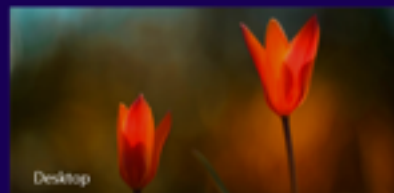
Internet Explorer



Help + Tips



SkyDrive



Desktop



Weather



Photos



Video



Music



Camera



Mint.com

New to the Store

Mint.com

Free Not yet rated

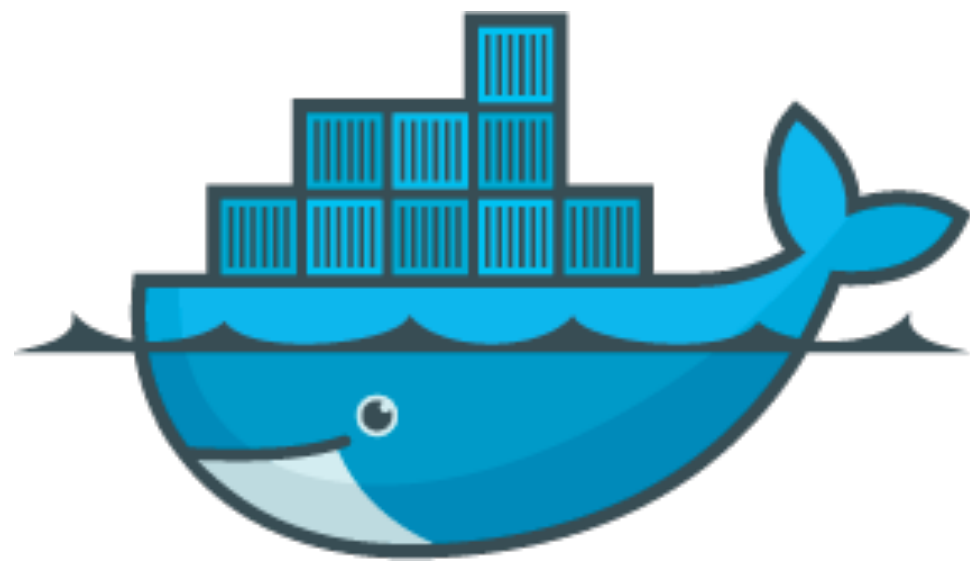


1

Football-Friendly: Chef
Marcus Samuelsson's...



Maps



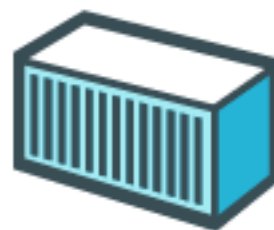
docker



An open platform for distributed applications for developers and sysadmins



Build



Ship

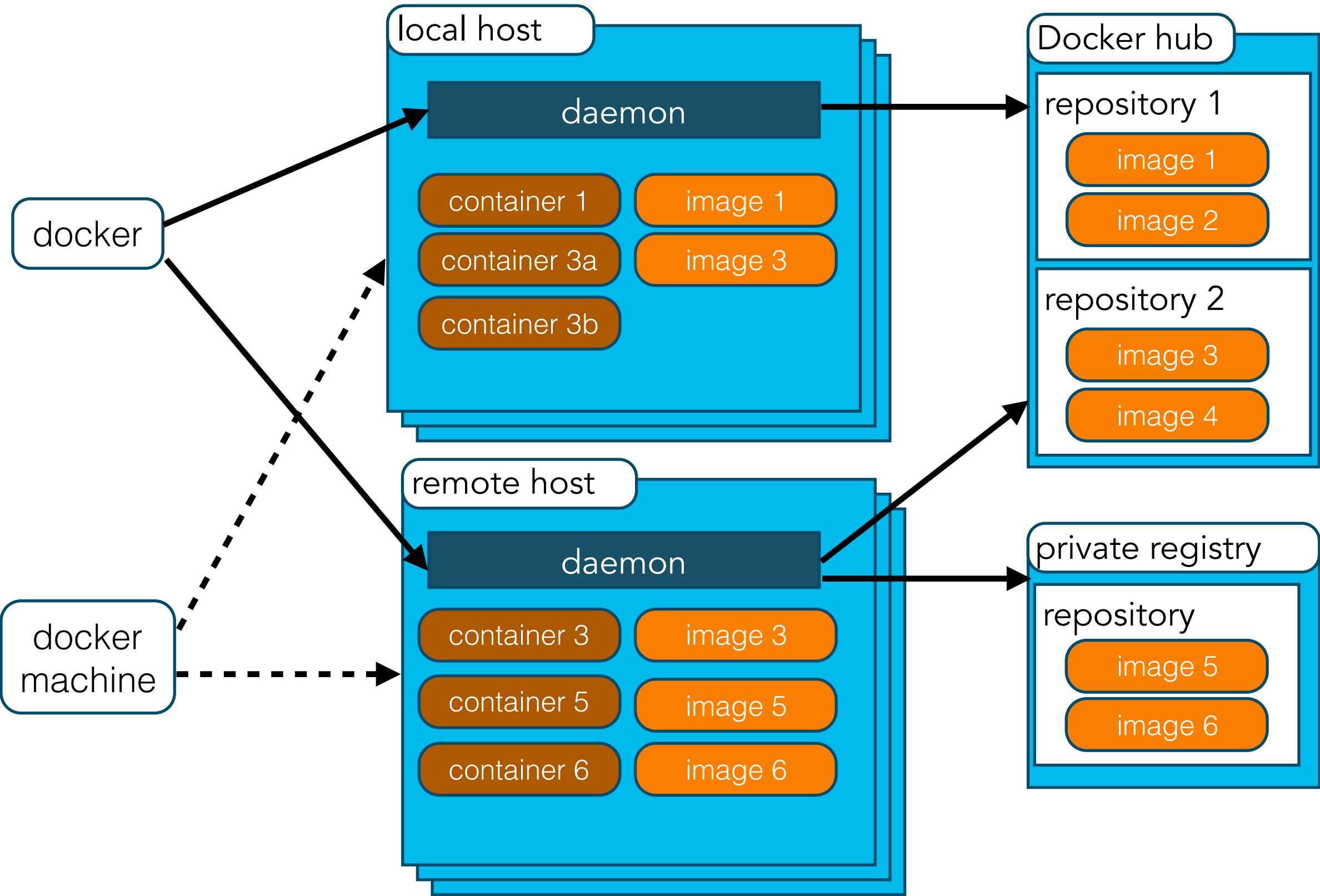


Run

Clients

Hosts

Registries





- 📌 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 /
```

```
COPY docker-entrypoint-initdb.d /docker-entrypoint-initdb.d/
```

```
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

client 1/project 1

daemon

jvm 1.7/tc 7

jvm 1.7/ tc 7

jvm 1.8/tc 8

jvm 1.8/tc 8

mysql 5.5

mysql 5.5

client 2/project 2

daemon

nginx 1.95

nginx 1.95

node 4.2.1

node 4.2.1

mongoDB 3

mongoDB 3

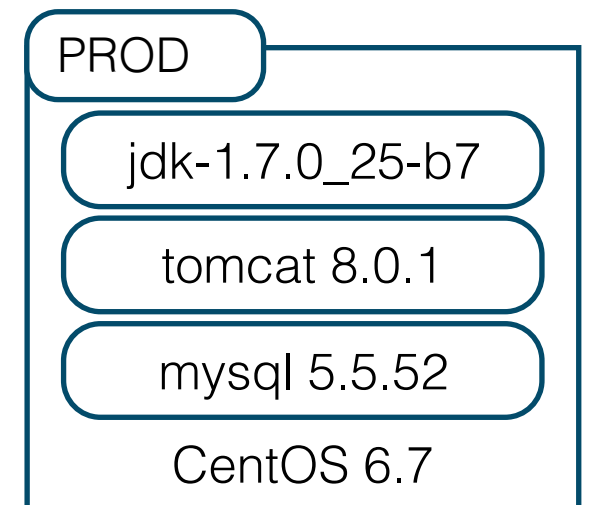
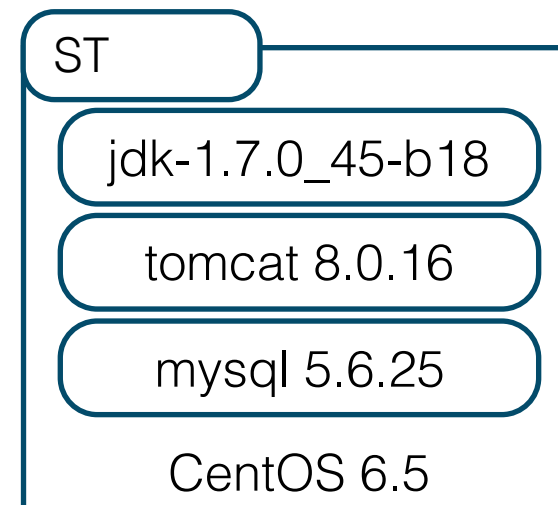
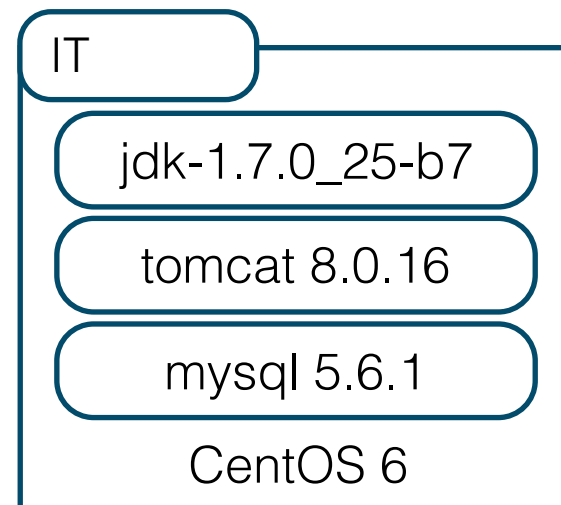
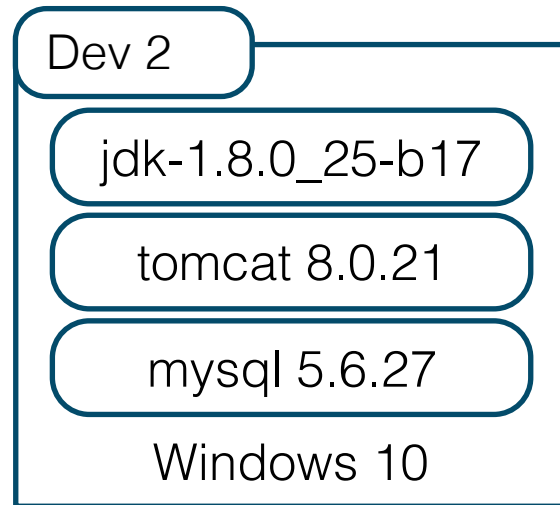
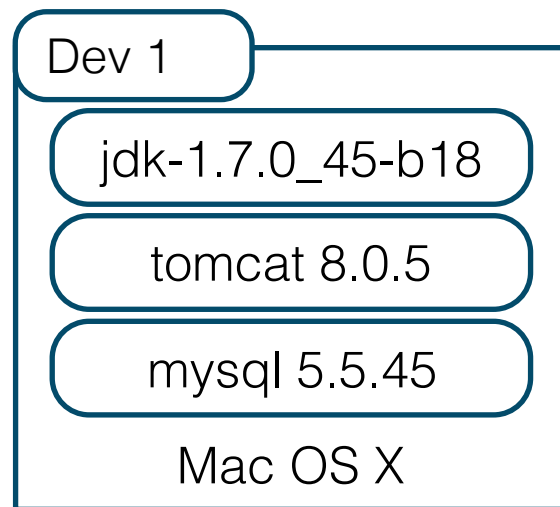
mysql 5.6

mysql 5.6

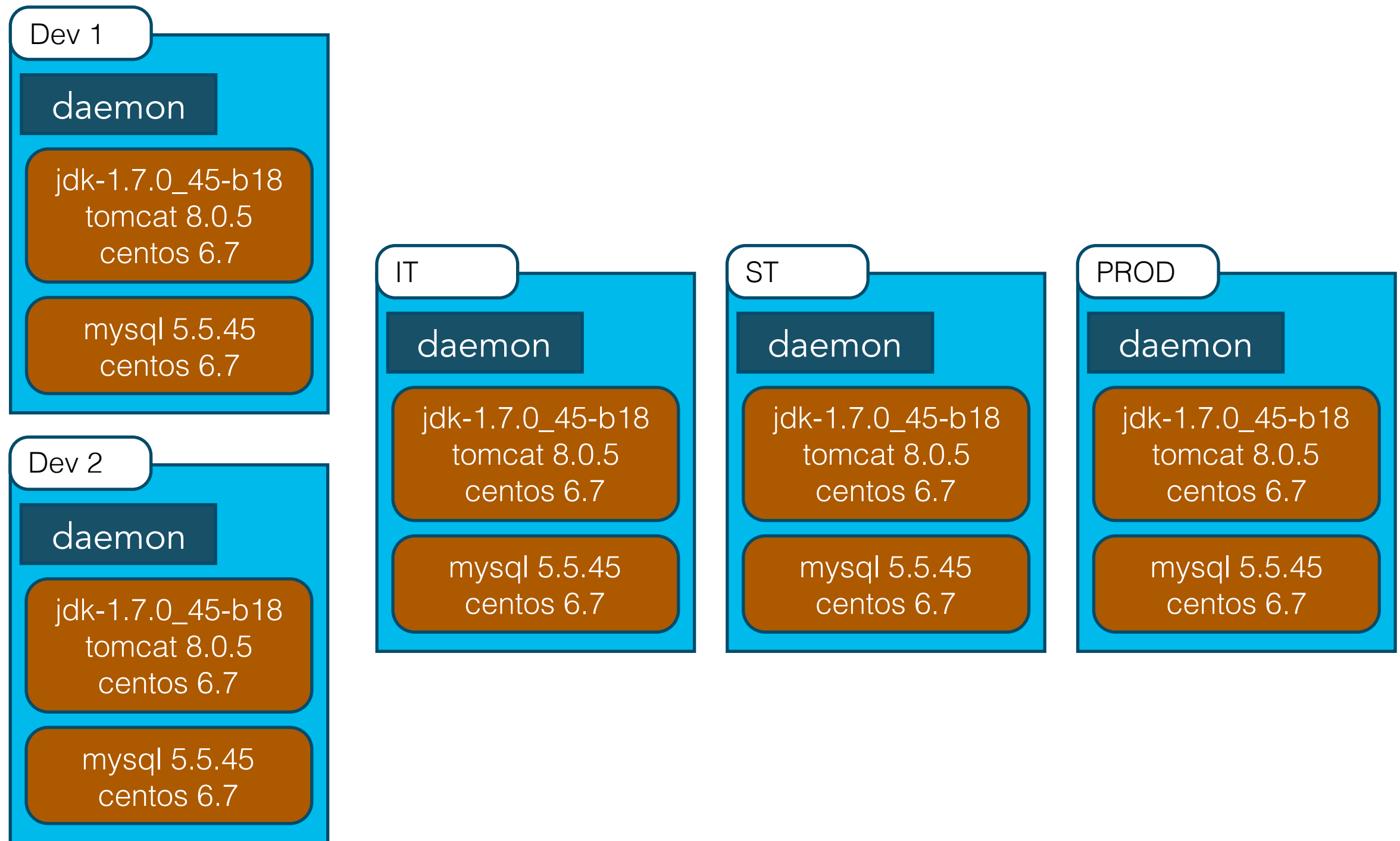
jvm 1.6

jvm 1.6

maintain consistency

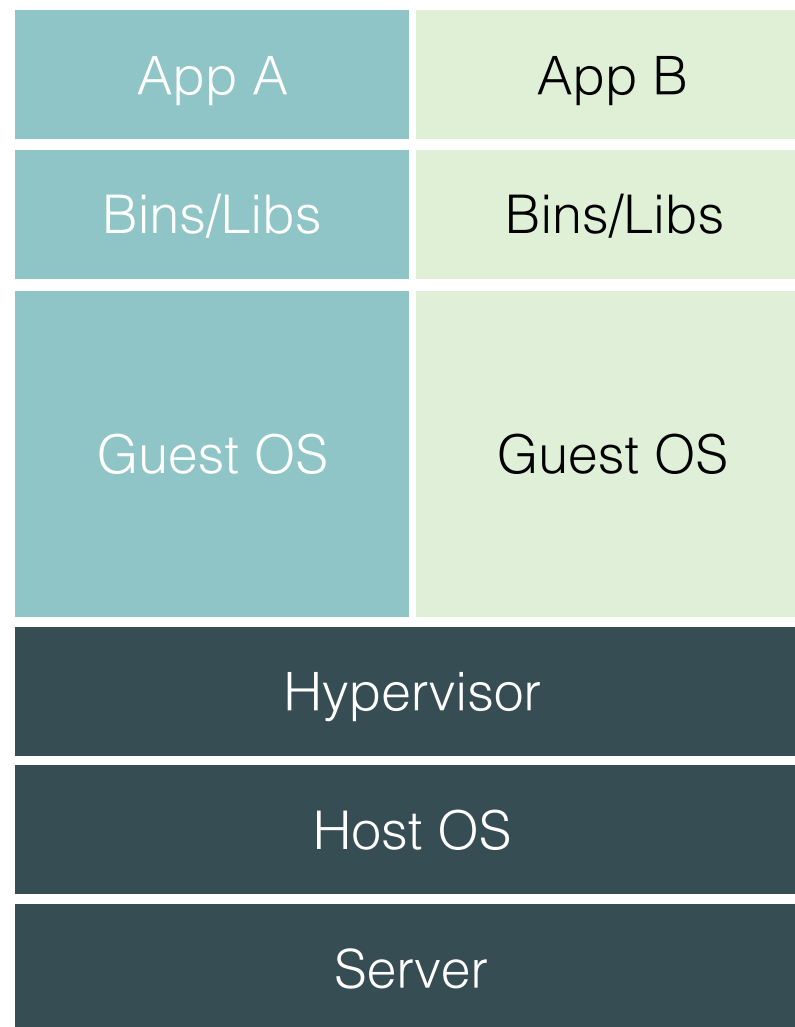


maintain consistency

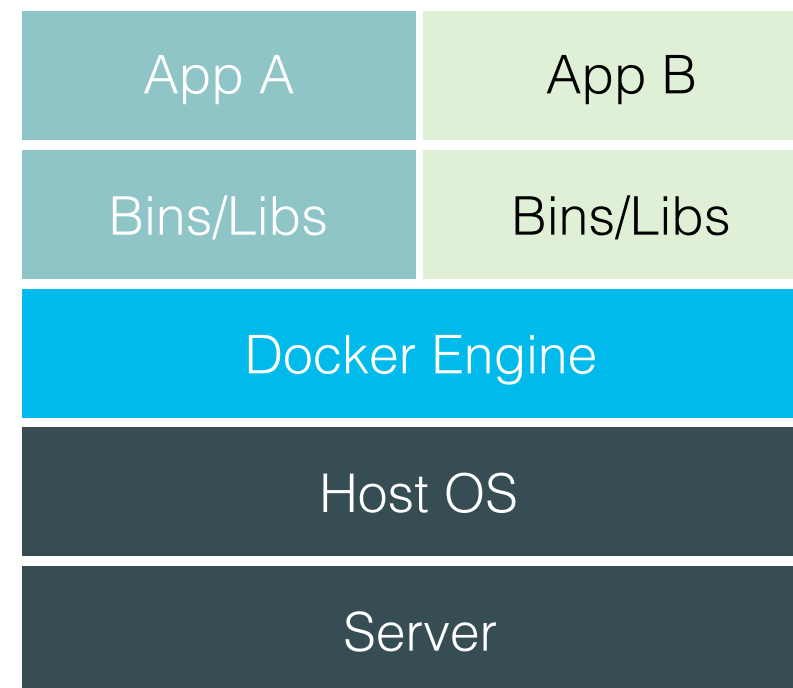


better resource utilization

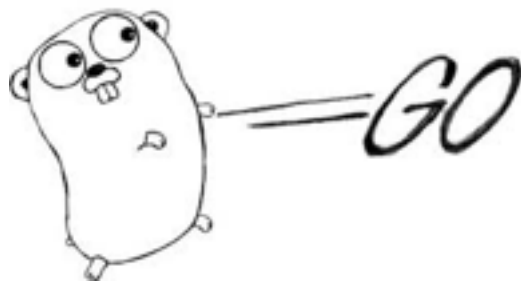
Virtual Machine



Docker



easy experimentation



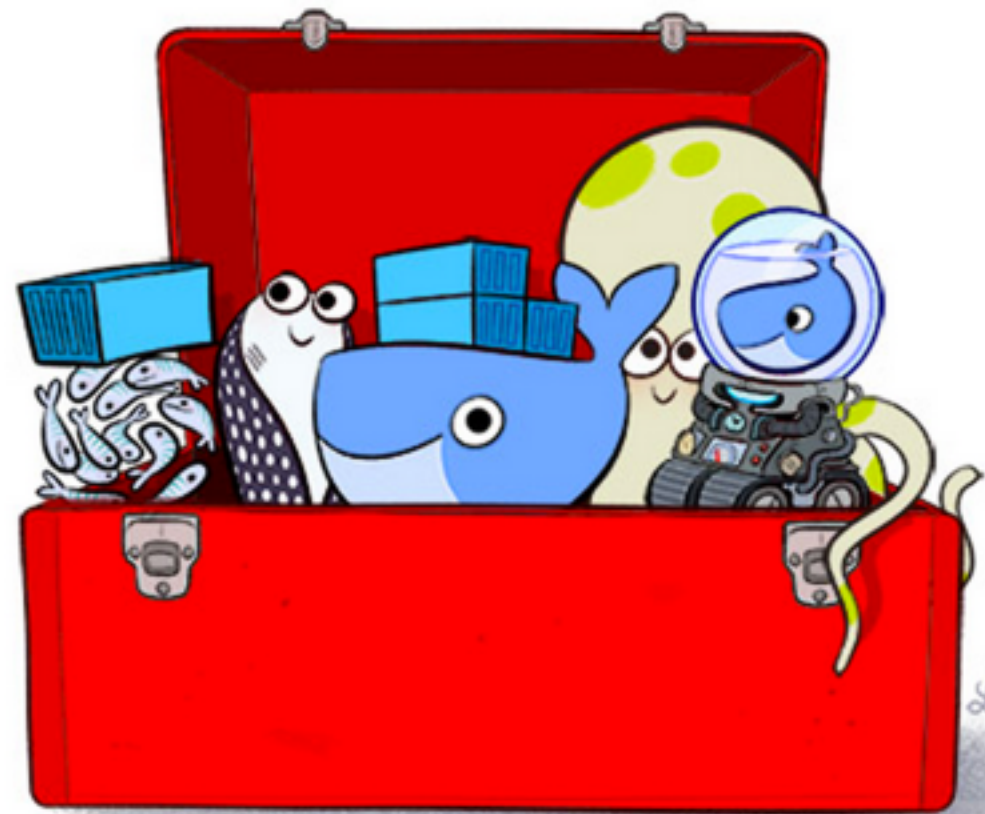
elasticsearch



SETUP DOCKER

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>



- 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`

Supported installation

Docker supports installation on the following:

- Amazon EC2 Installation
- Arch Linux
- Microsoft Azure platform
- Installation from binaries
- CentOS
- CRUX Linux
- Debian
- Fedora
- FrugalWare
- Gentoo
- Google Cloud Platform
- Install on Joyent Public Cloud
- Mac OS X
- Oracle Linux
- Rackspace Cloud
- Red Hat Enterprise Linux
- IBM SoftLayer
- openSUSE and SUSE Linux Enterprise
- Ubuntu
- Windows

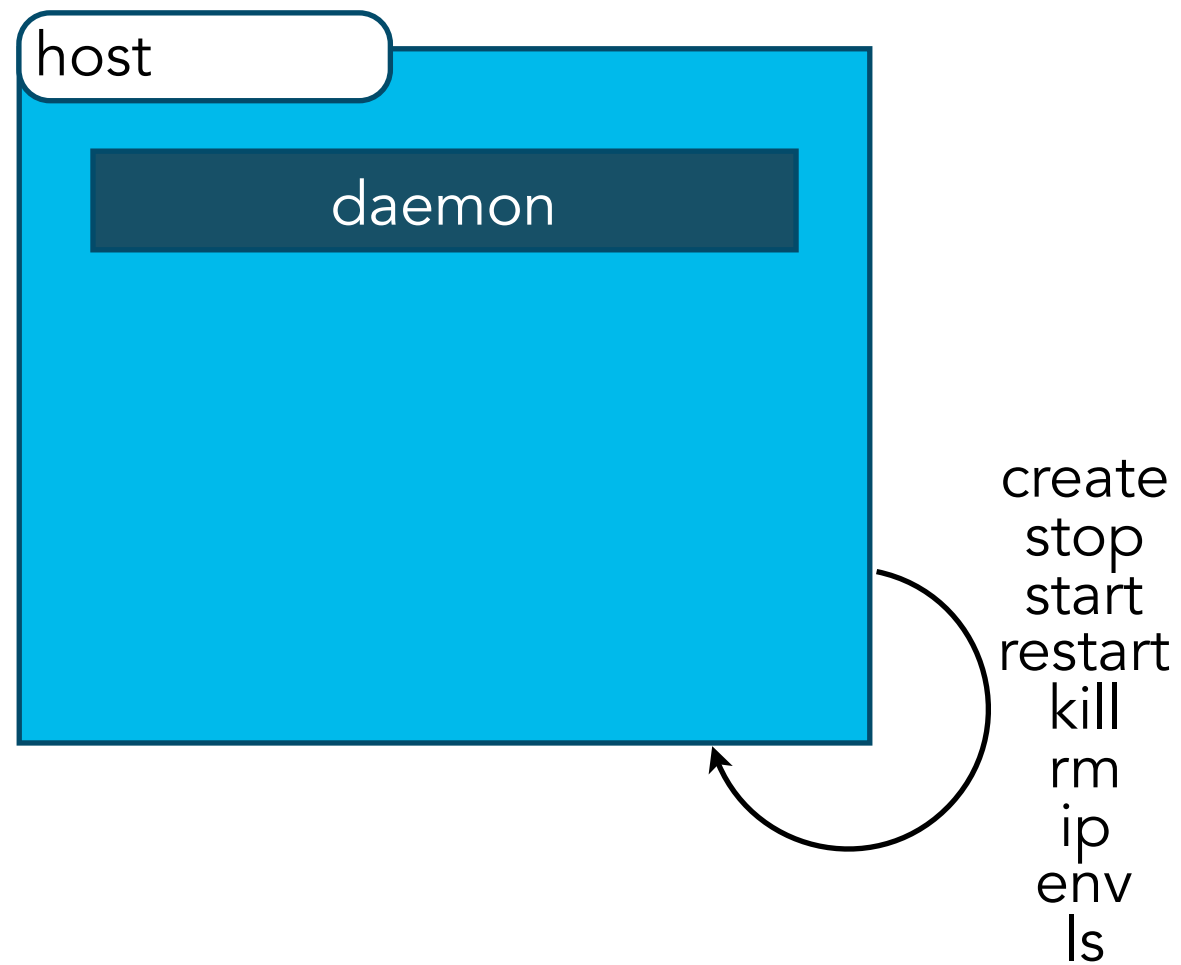
<https://docs.docker.com/installation/>

Lab I

I. Install Docker for your specific platform

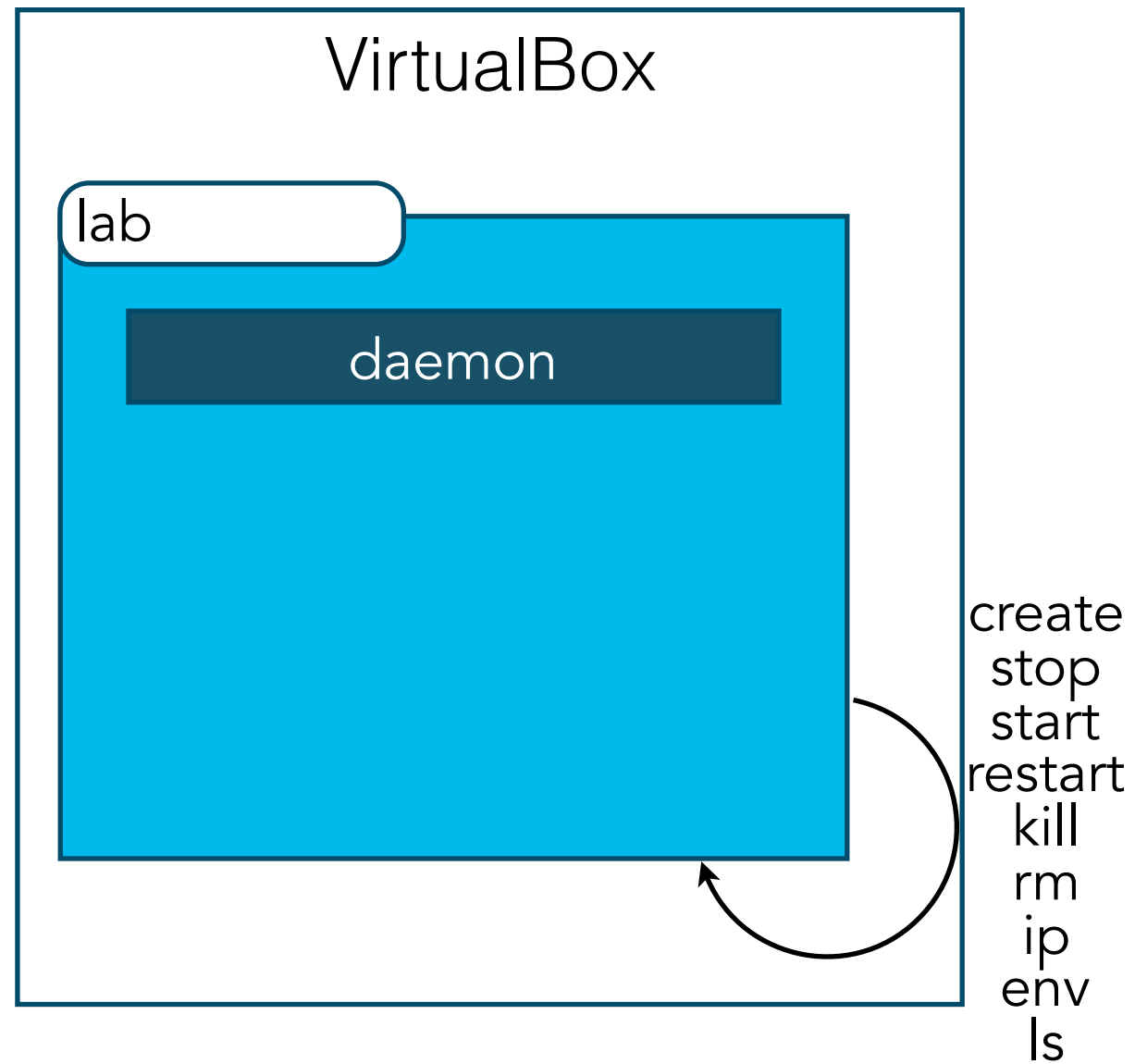
CREATE MACHINE

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					
client1		virtualbox	Running	tcp://192.168.99.100:2376	
client2					
lab		virtualbox	Running		

`docker-machine env lab`

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.102:2376"
export DOCKER_CERT_PATH="/Users/user"
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					
client2		virtualbox	Stopped		
lab	*	virtualbox	Running		





```
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

default
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

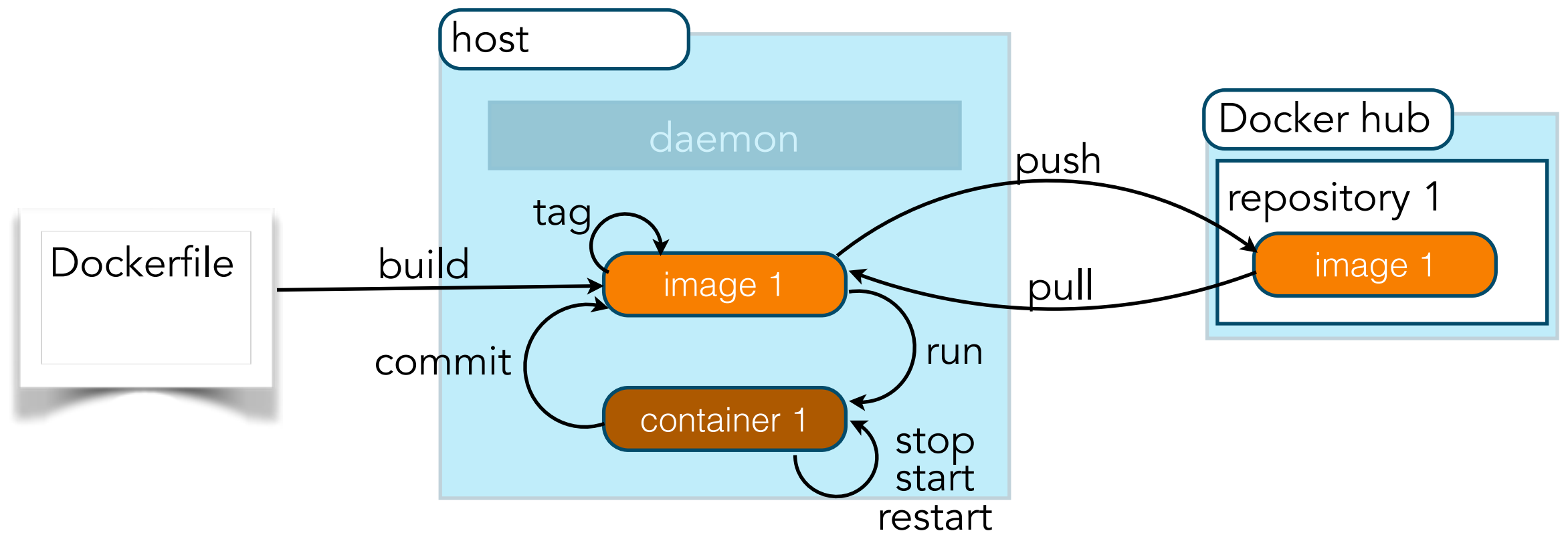
Adapter 1: Intel PRO/1000 MT Desktop (NAT)
Adapter 2: Intel PRO/1000 MT Desktop (Host-only Adapter, 'vboxnet0')

Lab 2

- I. Create a new Docker Machine called lab

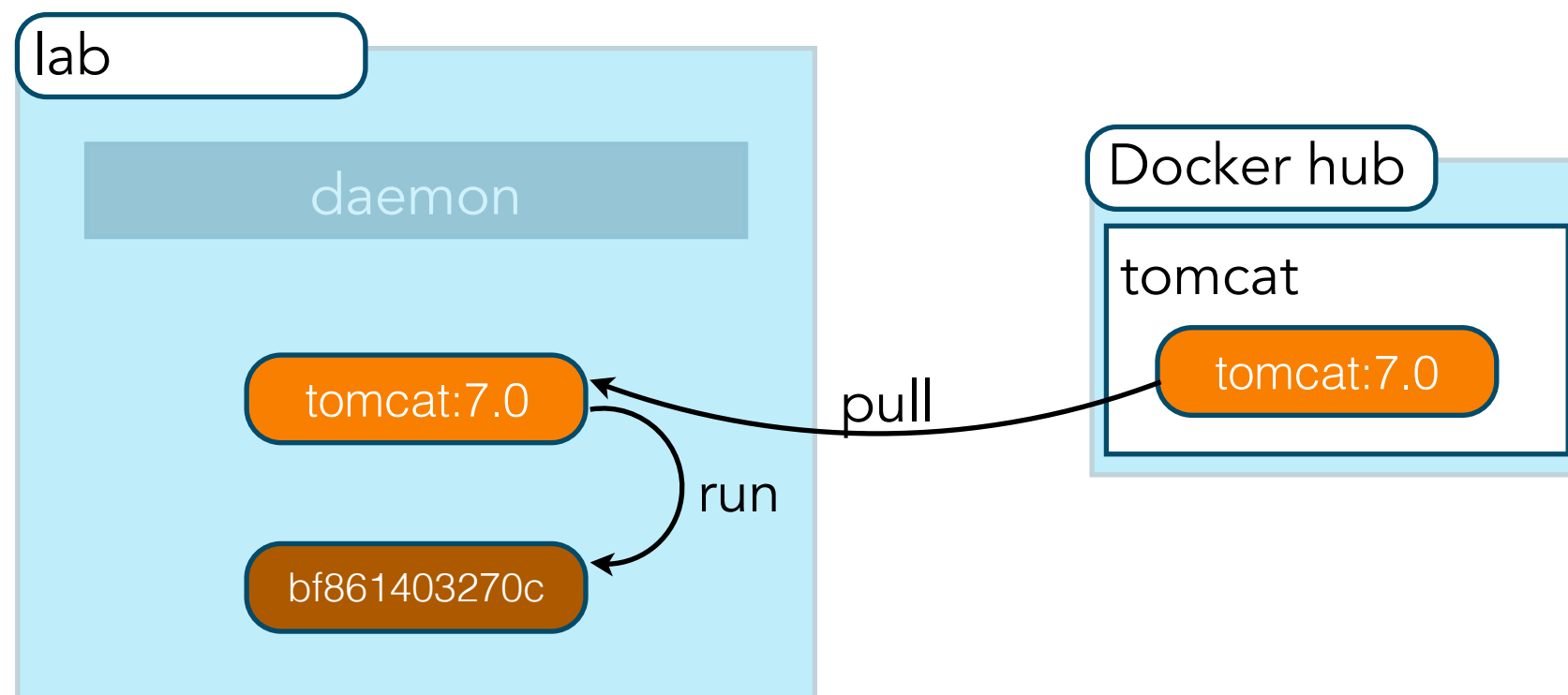
DOCKER LIFECYCLE

docker Lifecycle



`docker [OPTIONS] COMMAND [arg...]`

FINDING & RUNNING CONTAINERS





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/>



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](#)

([3 images for tomcat](#)). This image is updated via pull requests to [the docker](#)

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](#).

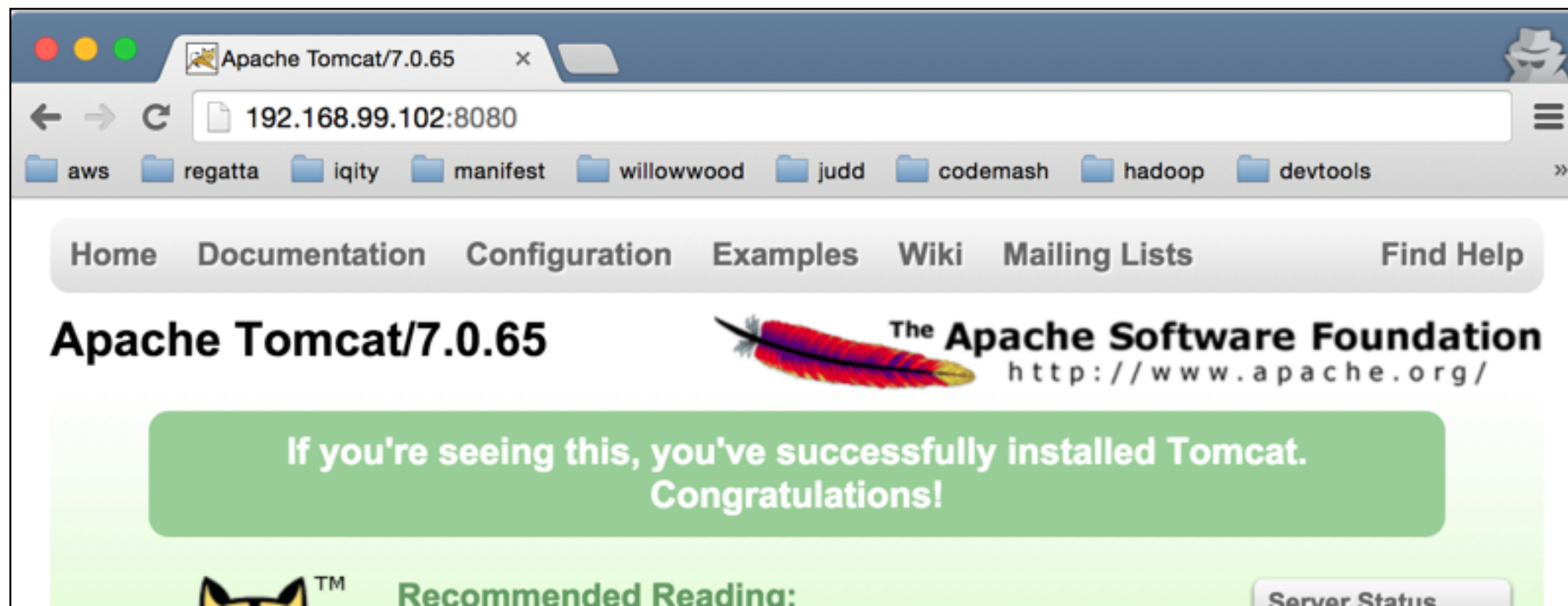
Contributing

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.

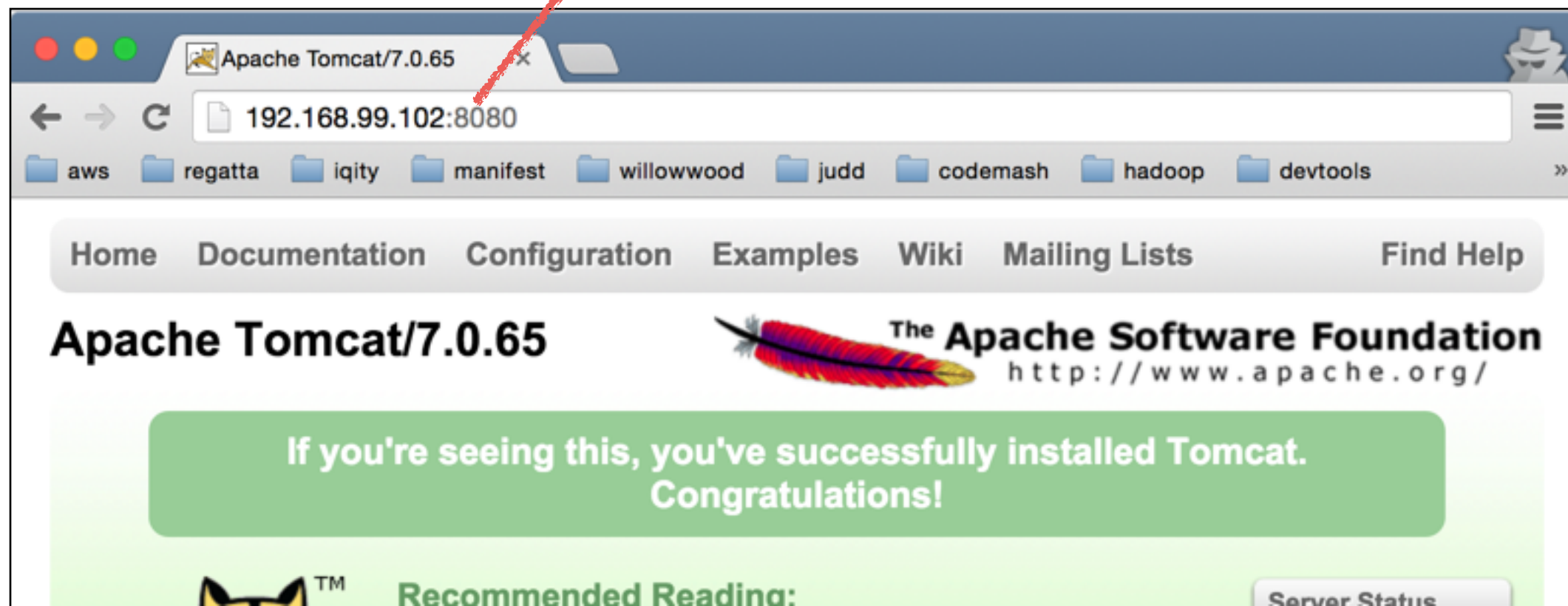
```
docker run -it --rm -p 8080:8080 tomcat:7.0-jre7
```

```
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
```



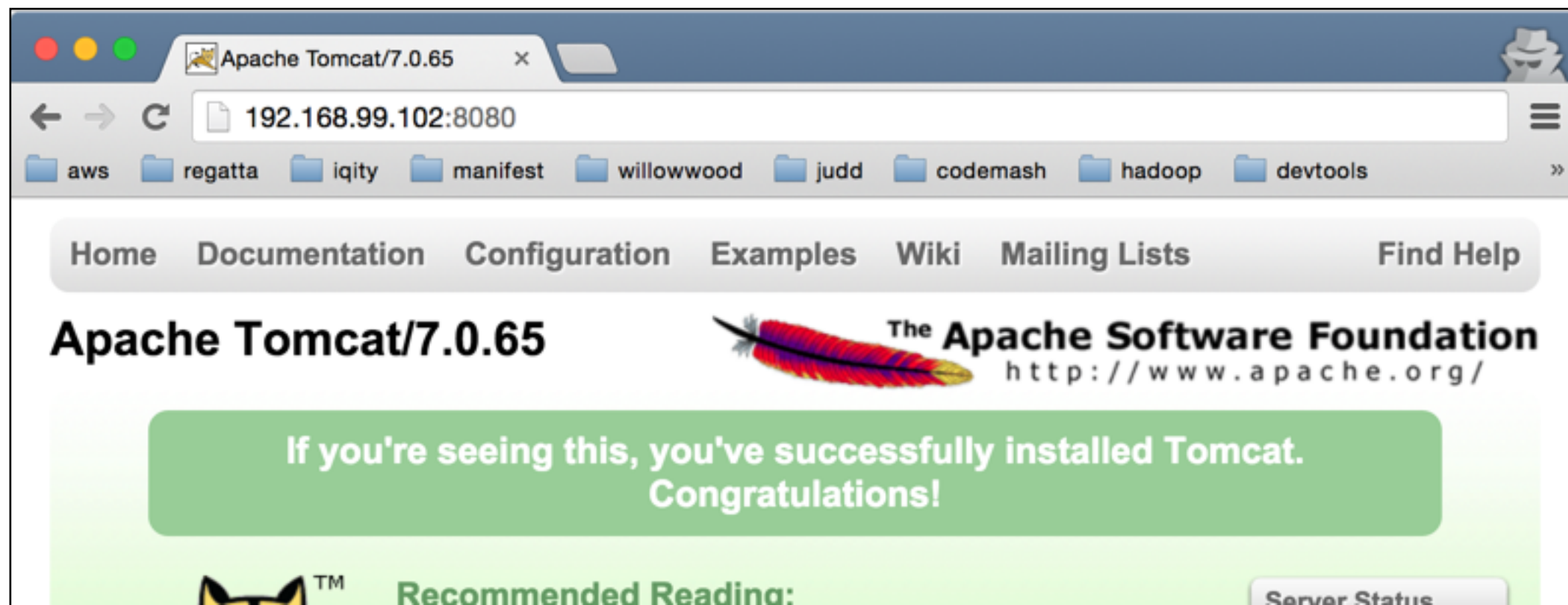
`docker run -it --rm -p 8080:8080 tomcat:7.0-jre7`

```
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 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: 4.0.0-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

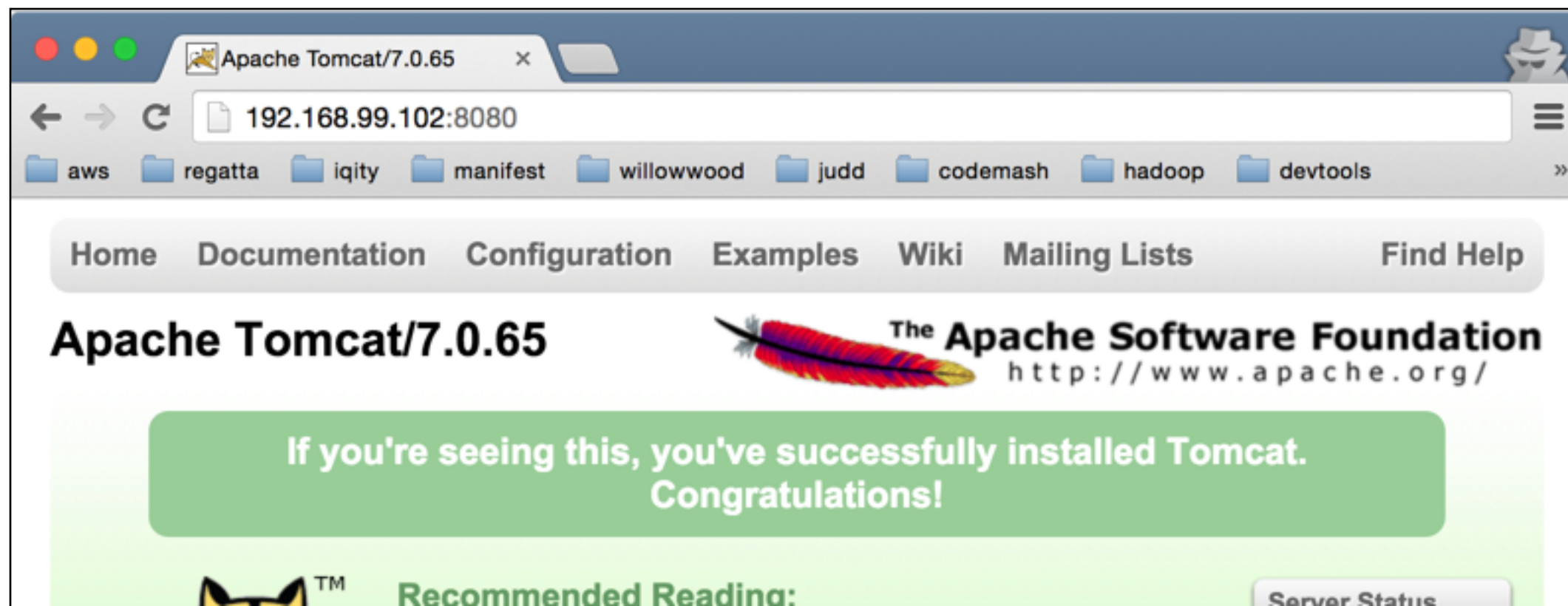


```
docker-machine ip lab
```

```
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	

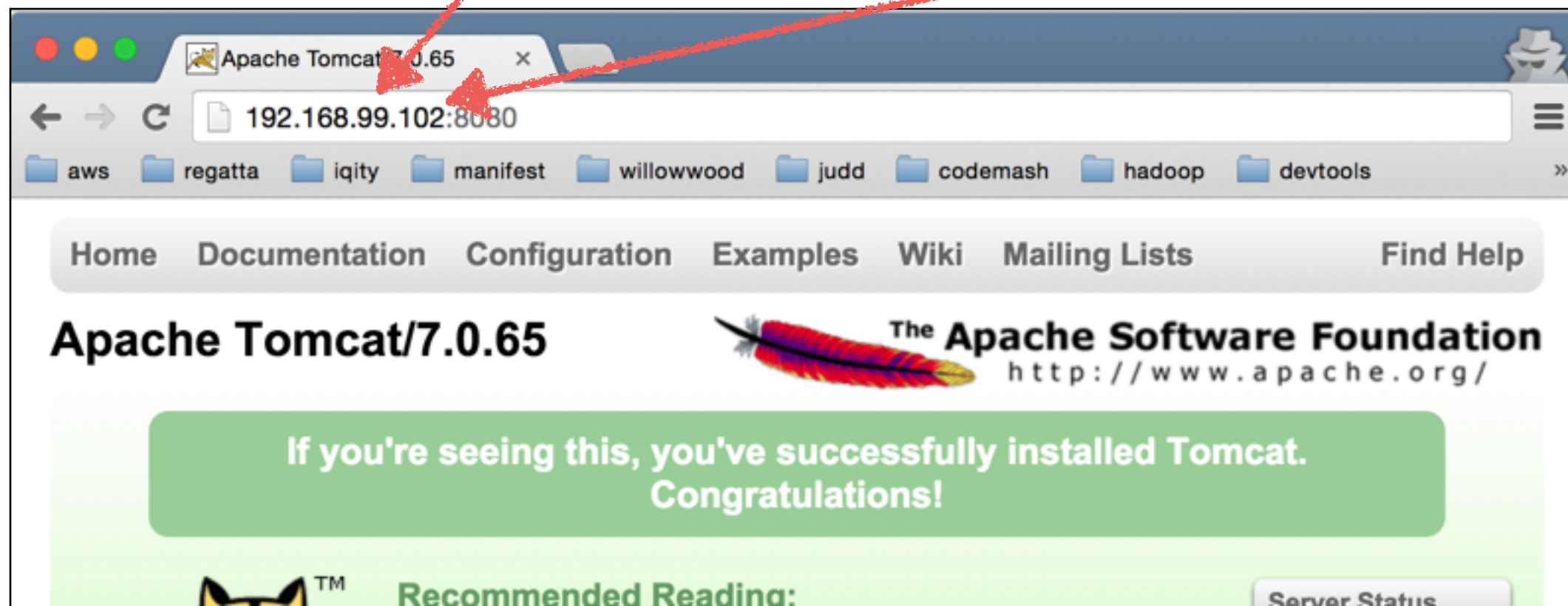


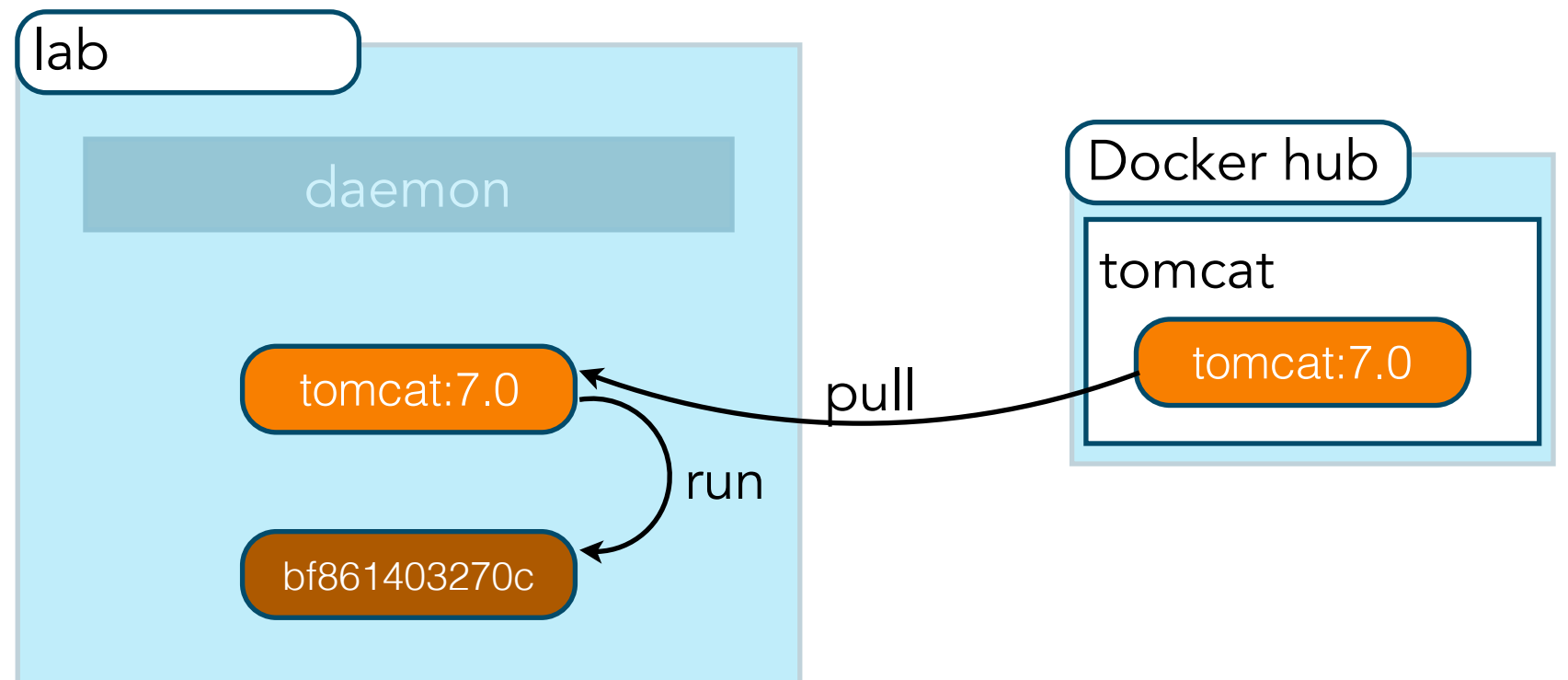
```
docker-machine ip lab
```

```
192.168.99.102
```

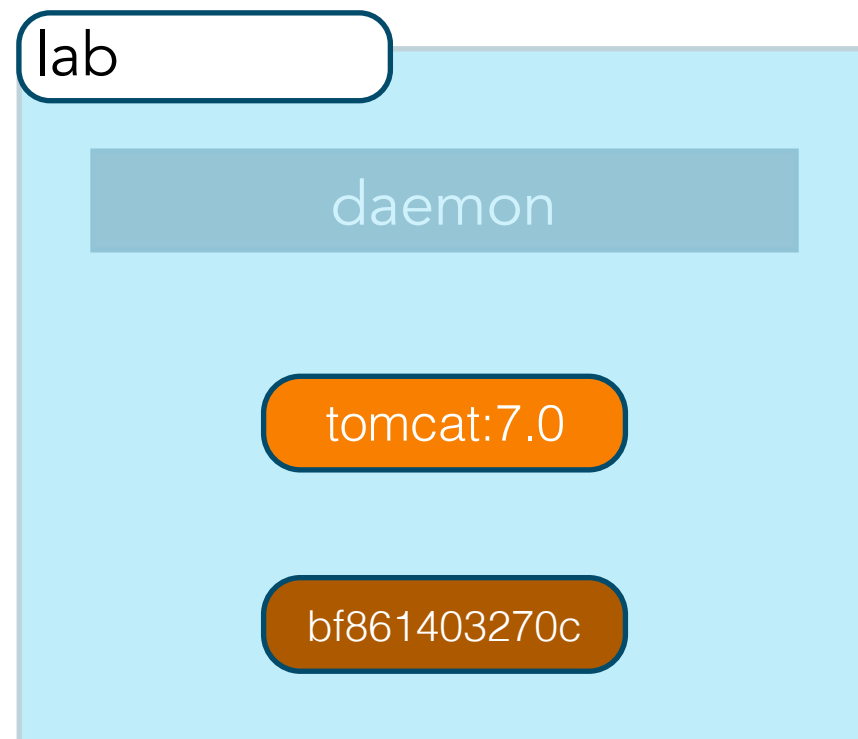
```
docker-machine ls
```

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	



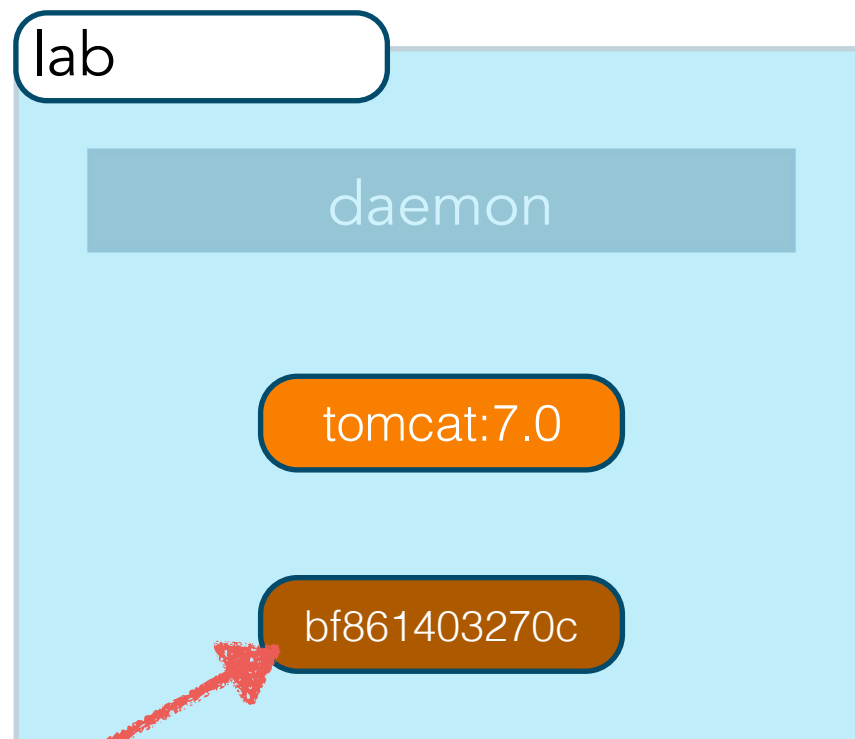


```
docker run -it --rm -p 8080:8080 tomcat:7.0-jre7
```



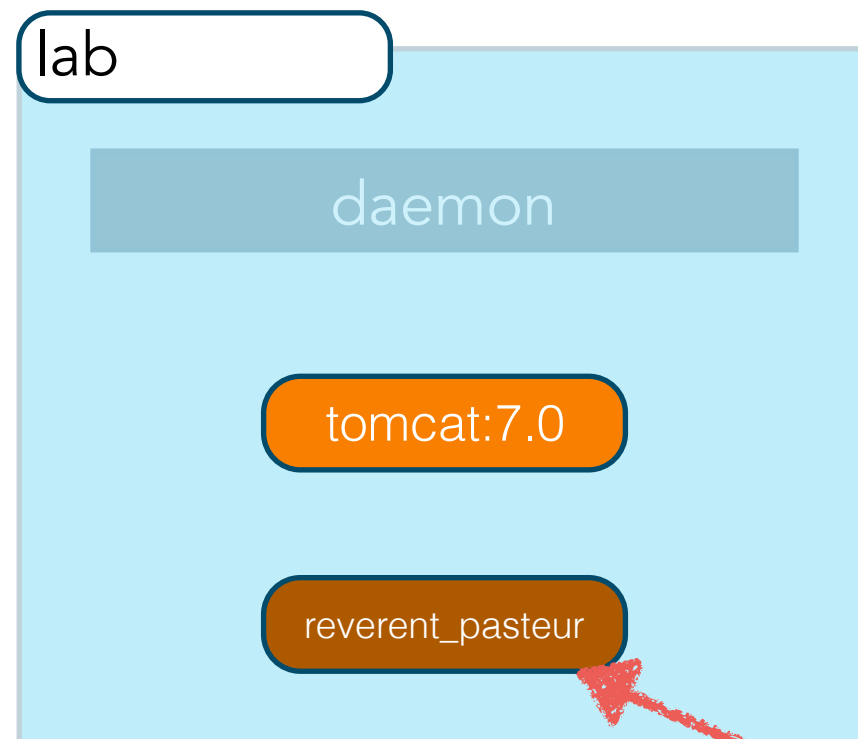
docker ps

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
```

```
COPY docker-entrypoint-initdb.d /docker-entrypoint-initdb.d/
```

```
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"]
```

docker-entrypoint-initdb.d/01-create-database.sql

```
CREATE DATABASE neuz;
```

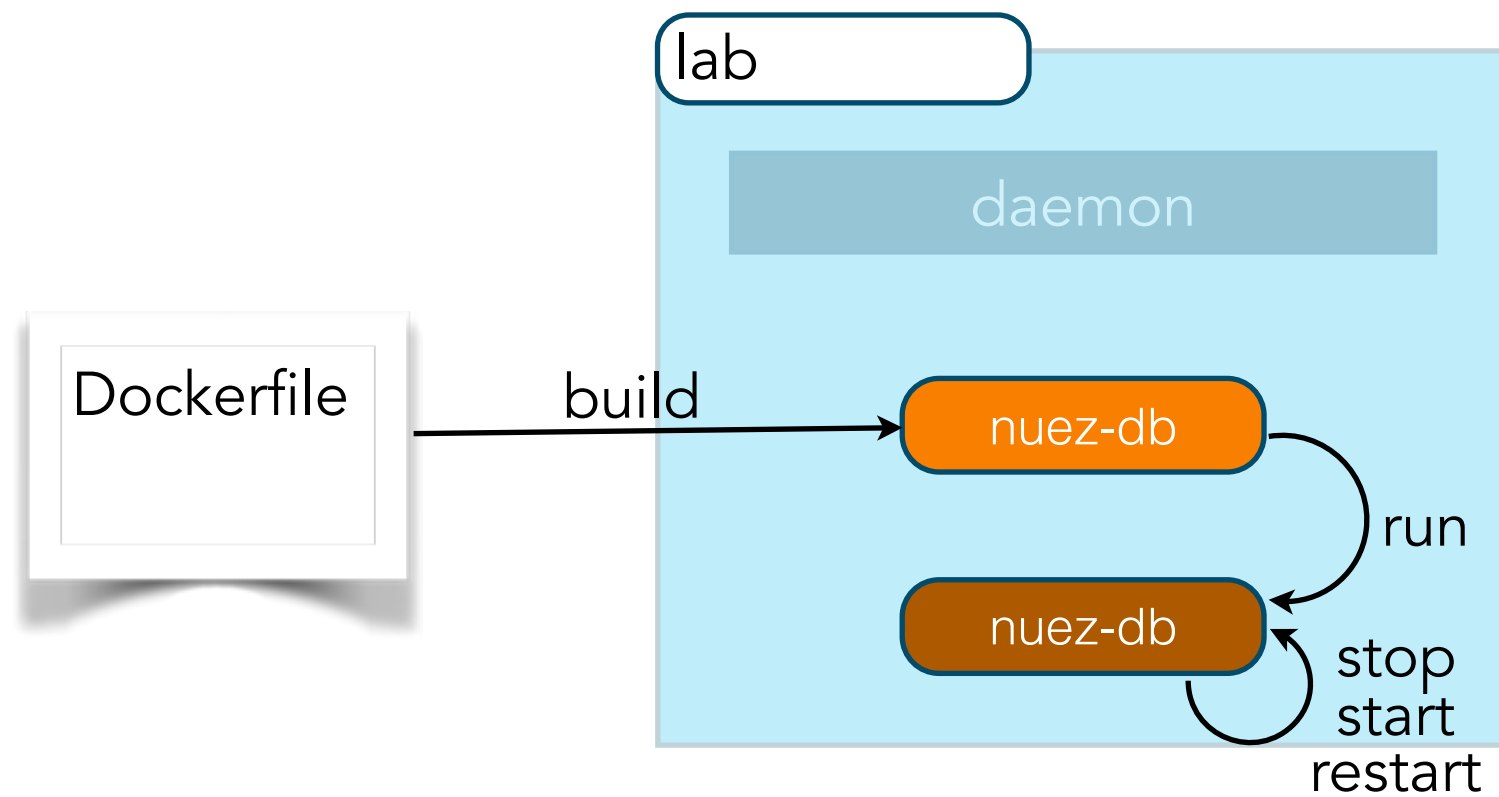
docker-entrypoint-initdb.d/02-create-users.sql

```
CREATE USER 'nuez'@'%' IDENTIFIED BY 'nuez+1';
```

```
GRANT ALL PRIVILEGES ON neuz.* TO 'nuez'@'%';
```

```
FLUSH privileges;
```

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 nuev-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: sha256:72a09a61824bdaf652e701fcfbf0ee12f5b132d8fdeaf1629ce42960375de03cb

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 bcb6b6f31fed3

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

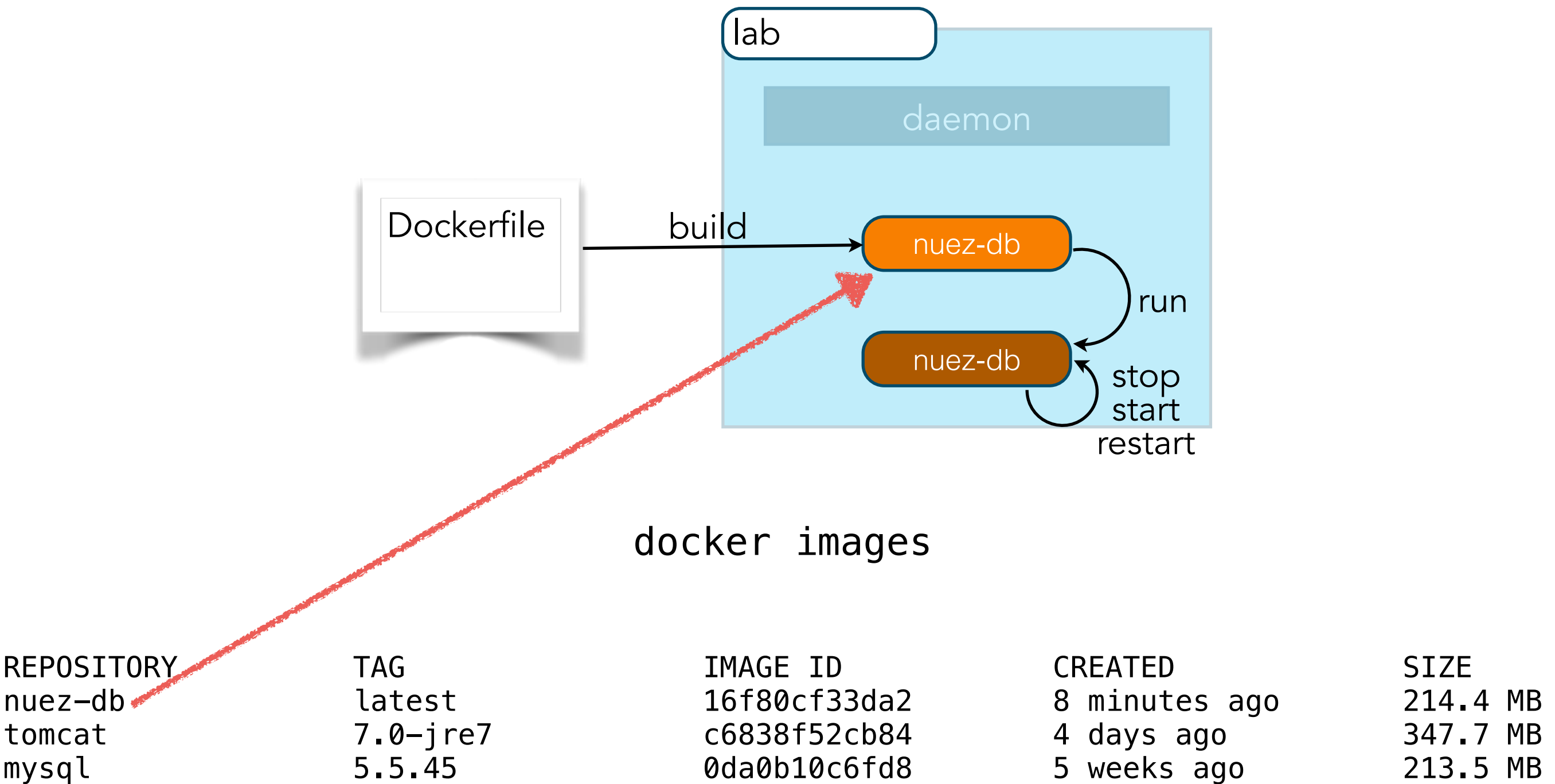
Step 5 : CMD mysqld

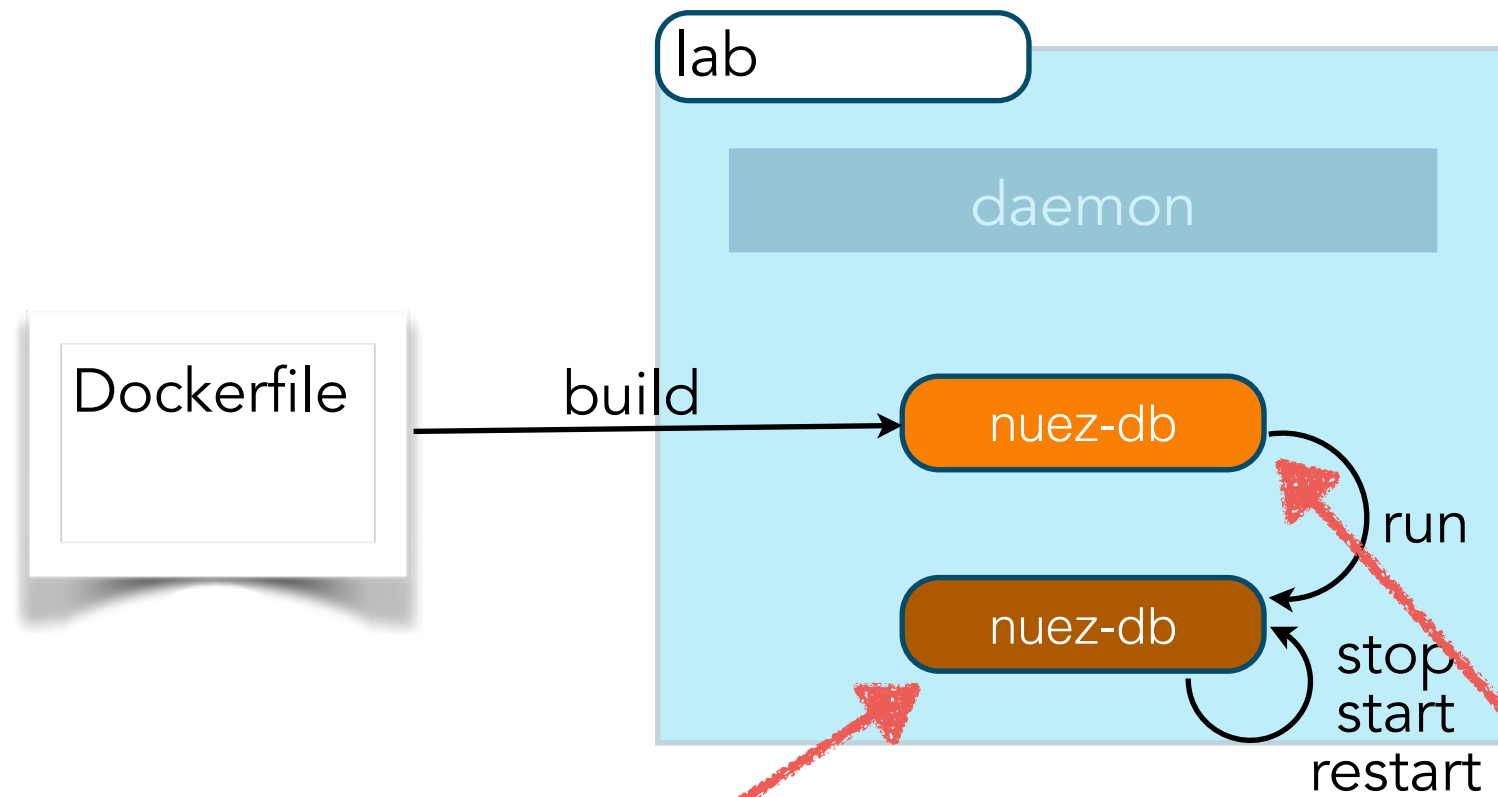
----> Running in 1cd40a7e38f8

----> 16f80cf33da2

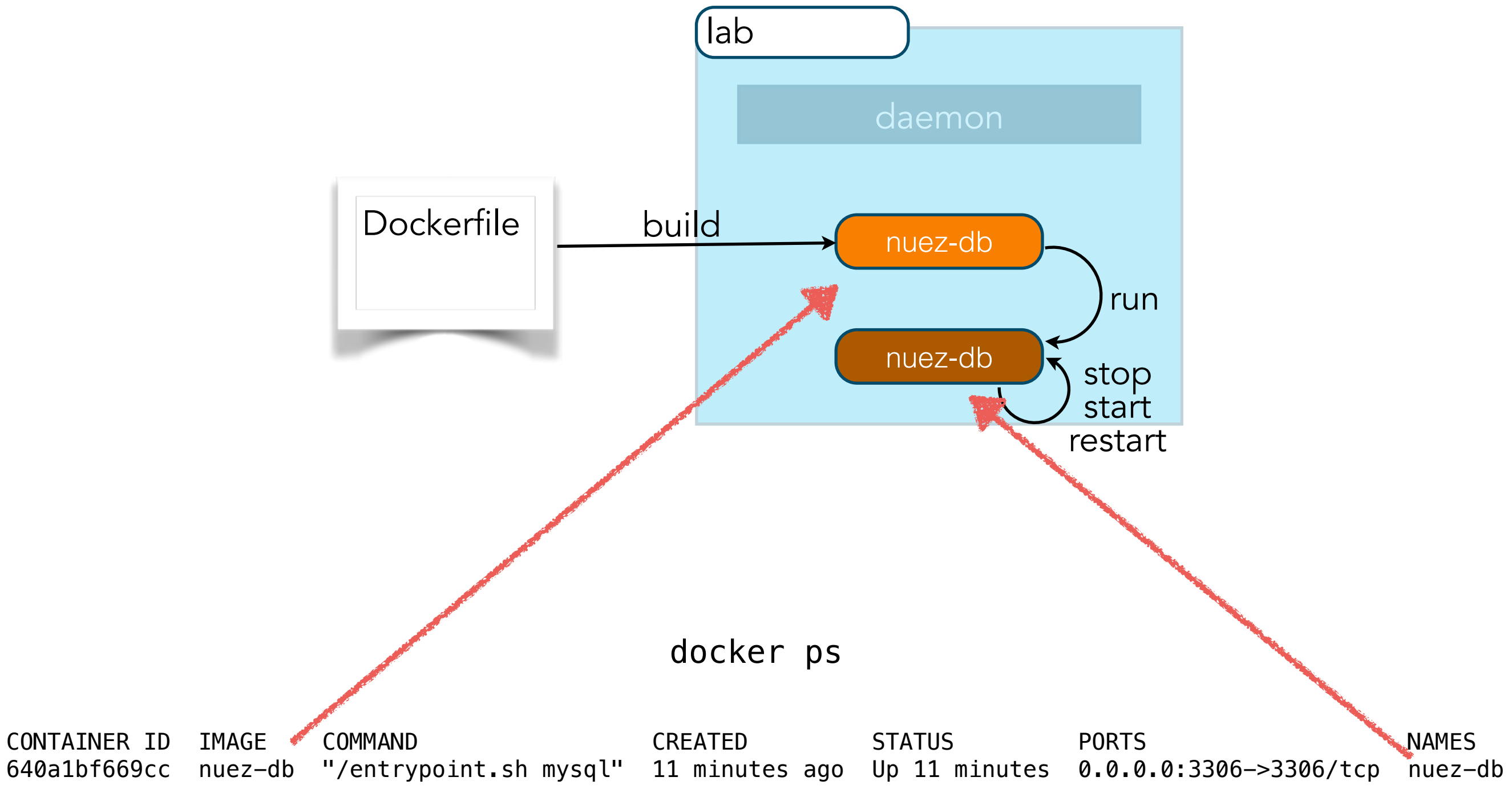
Removing intermediate container 1cd40a7e38f8

Successfully built 16f80cf33da2





```
docker 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
```



Lab 4

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

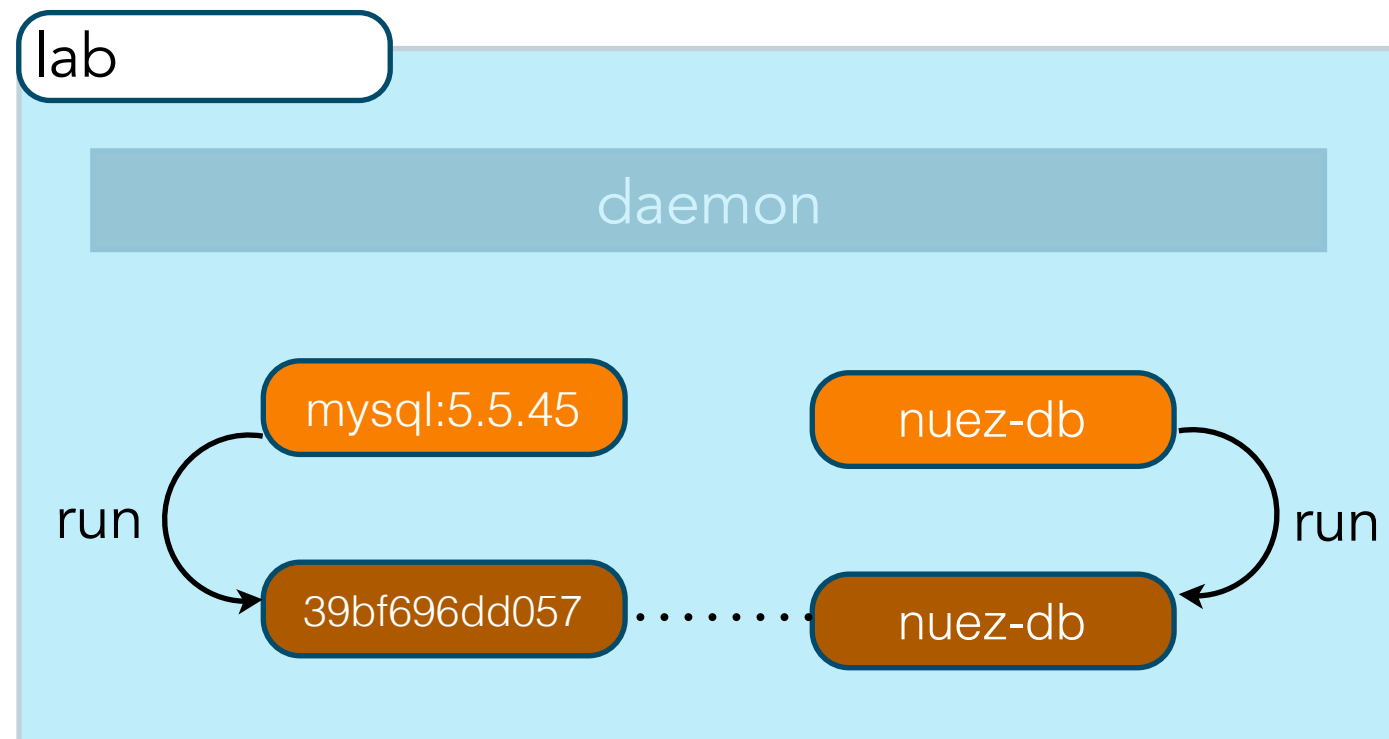
LINKING TO CONTAINER

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

or

```
docker run -it --link nuev-db:mysql --rm mysql:5.5.45 bash
```

```
mysql -h $MYSQL_PORT_3306_TCP_ADDR -u $MYSQL_ENV_MYSQL_USER -p$MYSQL_ENV_MYSQL_PASSWORD $MYSQL_ENV_MYSQL_DATABASE
```



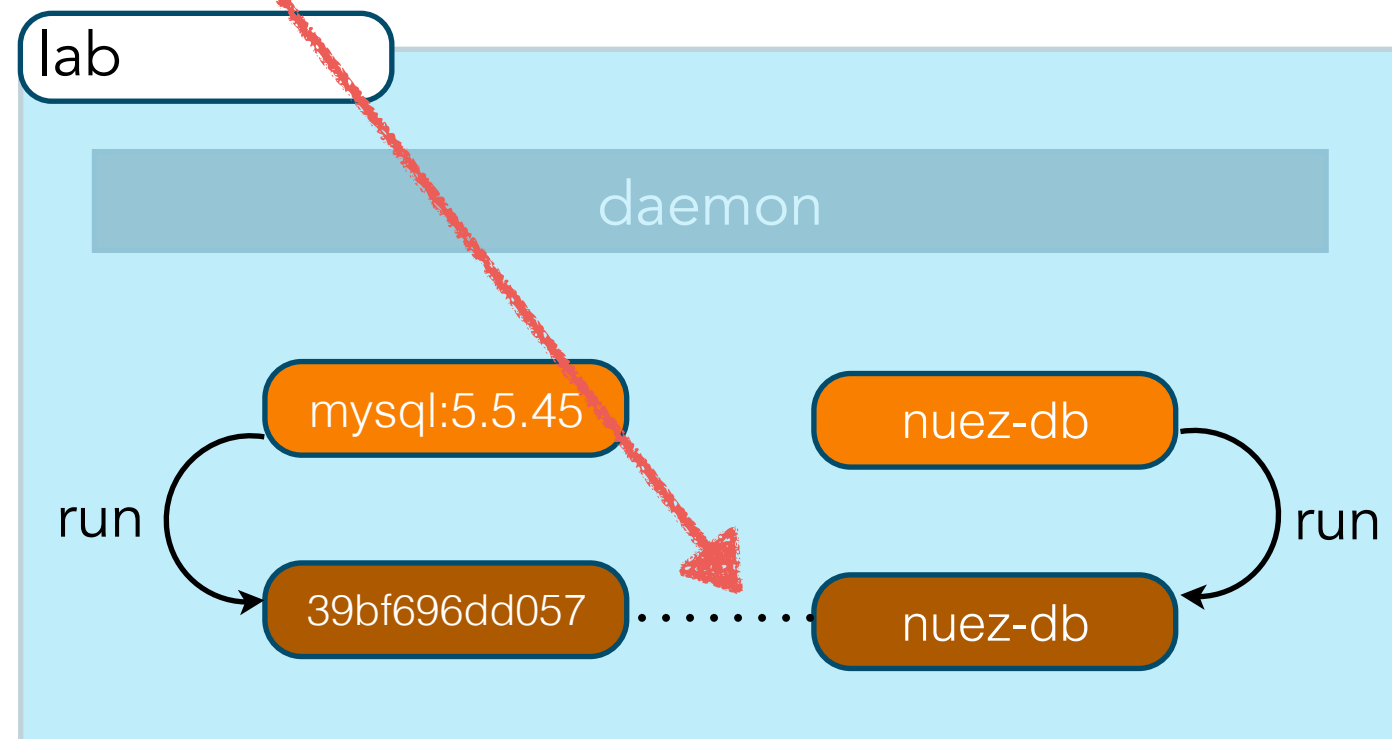
```
mysql -h 192.168.99.102 -u root -proot+1 nuev
```

or

```
docker run -it --link nuev-db:mysql --rm mysql:5.5.45 bash
```

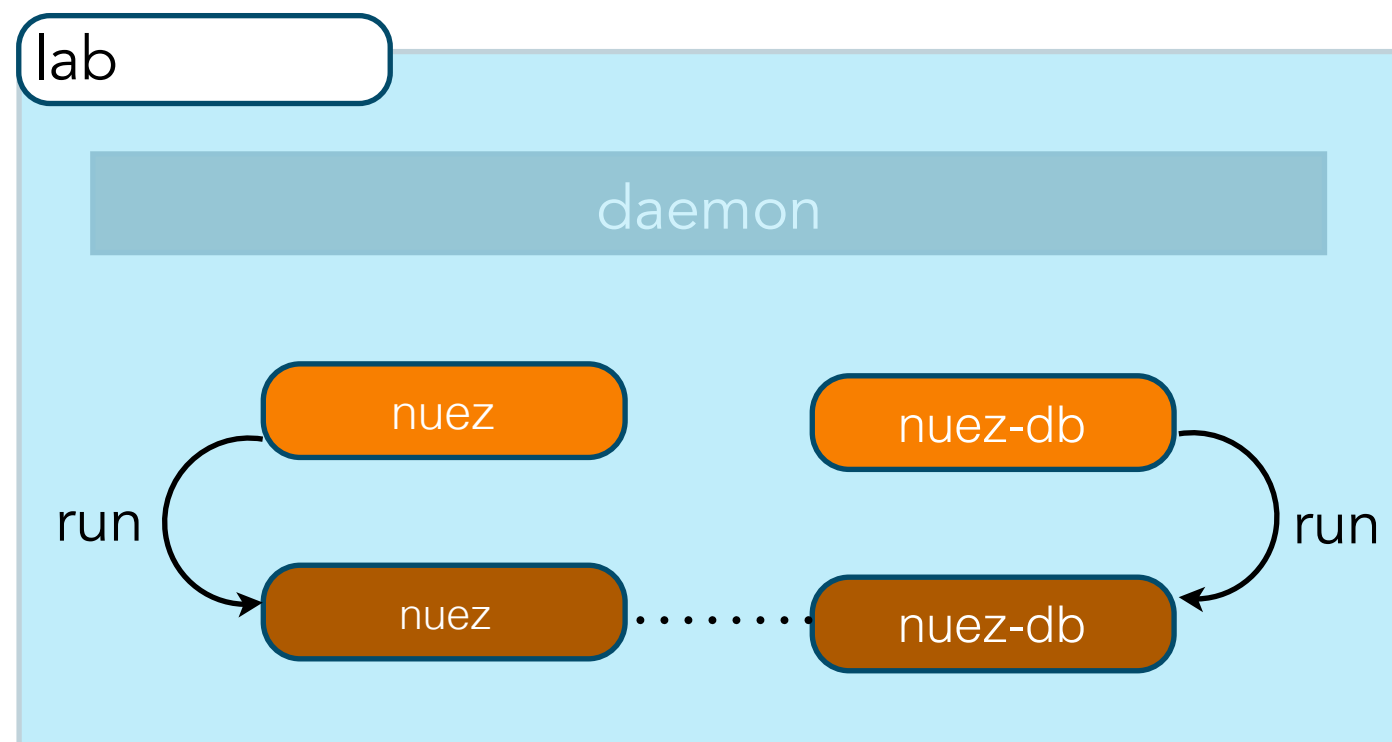
```
mysql -h $MYSQL_PORT_3306_TCP_ADDR -u $MYSQL_ENV_MYSQL_USER -p$MYSQL_ENV_MYSQL_PASSWORD $MYSQL_ENV_MYSQL_DATABASE
```

```
HOSTNAME=05f85a8745c2
MYSQL_ENV_MYSQL_DATABASE=nuev
MYSQL_ENV_MYSQL_ROOT_PASSWORD=root+1
TERM=xterm
MYSQL_VERSION=5.5.45
MYSQL_PORT_3306_TCP_PORT=3306
MYSQL_PORT_3306_TCP=tcp://172.17.0.28:3306
PATH=/usr/local/sbin:/usr/local/bin
MYSQL_ENV_MYSQL_USER=nuev-app
PWD=/
MYSQL_ENV_MYSQL_PASSWORD=nuev+1
MYSQL_ENV_MYSQL_VERSION=5.5.45
HOME=/root
SHLVL=1
MYSQL_PORT_3306_TCP_PROTO=tcp
MYSQL_NAME=/elated_franklin/mysql
MYSQL_MAJOR=5.5
MYSQL_PORT_3306_TCP_ADDR=172.17.0.28
MYSQL_ENV_MYSQL_MAJOR=5.5
MYSQL_PORT=tcp://172.17.0.28:3306
_=/usr/bin/env
```



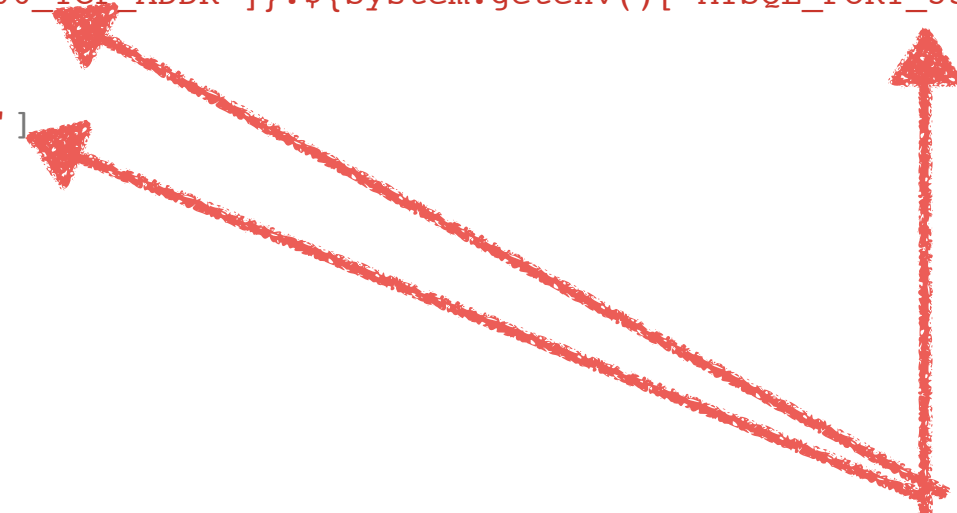
Lab 5

- I. Run a mysql command-line in a container and connect to the nuev-db.



```
// remove for brevity
```

```
docker {  
  dataSource {  
    dbCreate = "update"  
    url = "jdbc:mysql://${System.getenv()['MYSQL_PORT_3306_TCP_ADDR']}:${System.getenv()['MYSQL_PORT_3306_TCP_PORT']}/nuez"  
    driverClassName = "com.mysql.jdbc.Driver"  
    username = System.getenv()['MYSQL_ENV_MYSQL_USER']  
    password = System.getenv()['MYSQL_ENV_MYSQL_PASSWORD']  
    pooled = true  
    properties {  
      maxActive = -1  
      minEvictableIdleTimeMillis=1800000  
      timeBetweenEvictionRunsMillis=1800000  
      numTestsPerEvictionRun=3  
      testOnBorrow=true  
      testWhileIdle=true  
      testOnReturn=true  
      validationQuery="SELECT 1"  
    }  
  }  
}  
// remove for brevity
```



Three red arrows originate from a common point on the right side of the image and point to the environment variable placeholders in the code: `'MYSQL_PORT_3306_TCP_ADDR'`, `'MYSQL_PORT_3306_TCP_PORT'`, and `'MYSQL_ENV_MYSQL_PASSWORD'`.

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-clouddevelop-2015/nuez-0.1.war  
    $CATALINA_HOME/webapps/ROOT.war
```

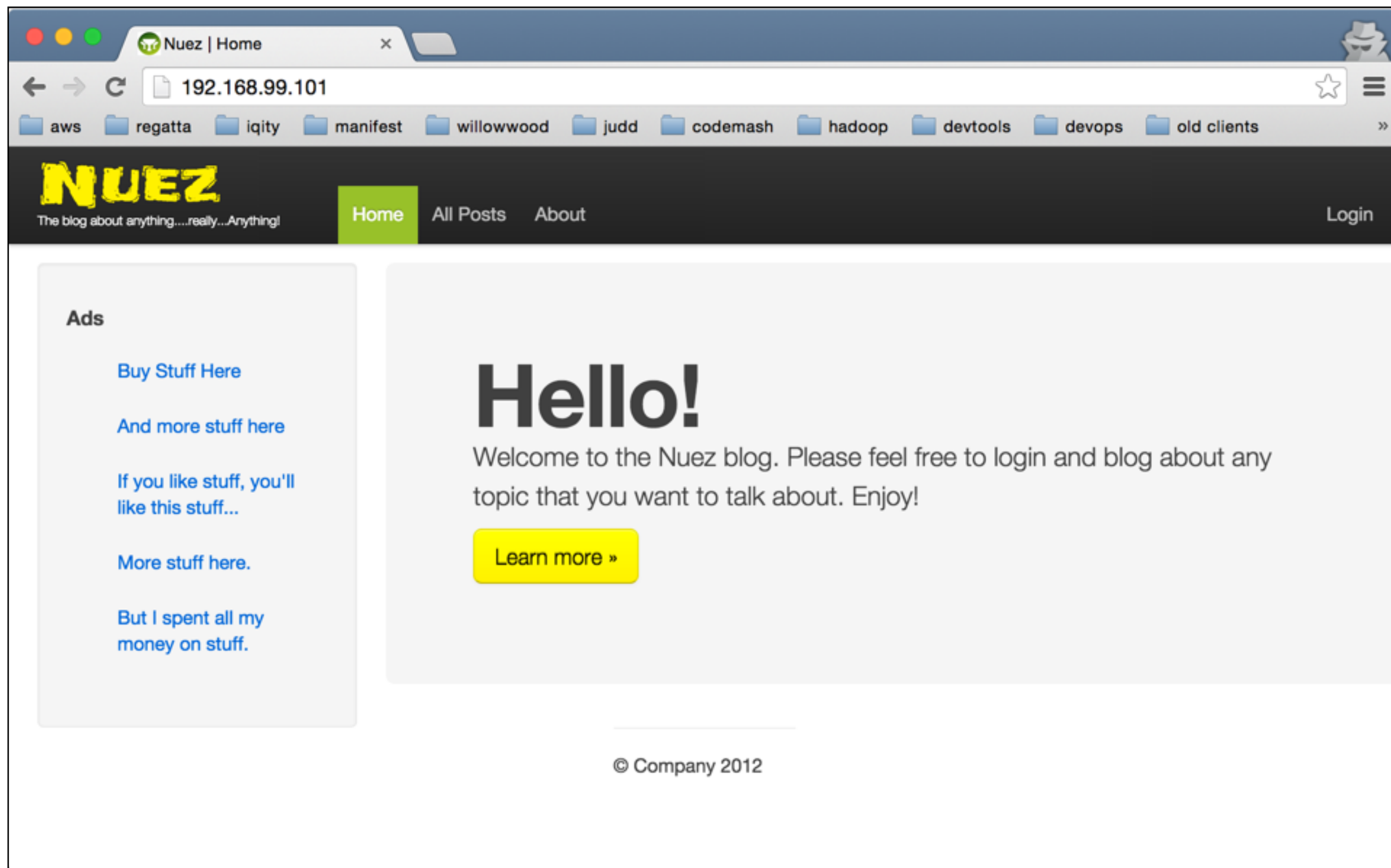
```
CMD ["catalina.sh", "run"]
```

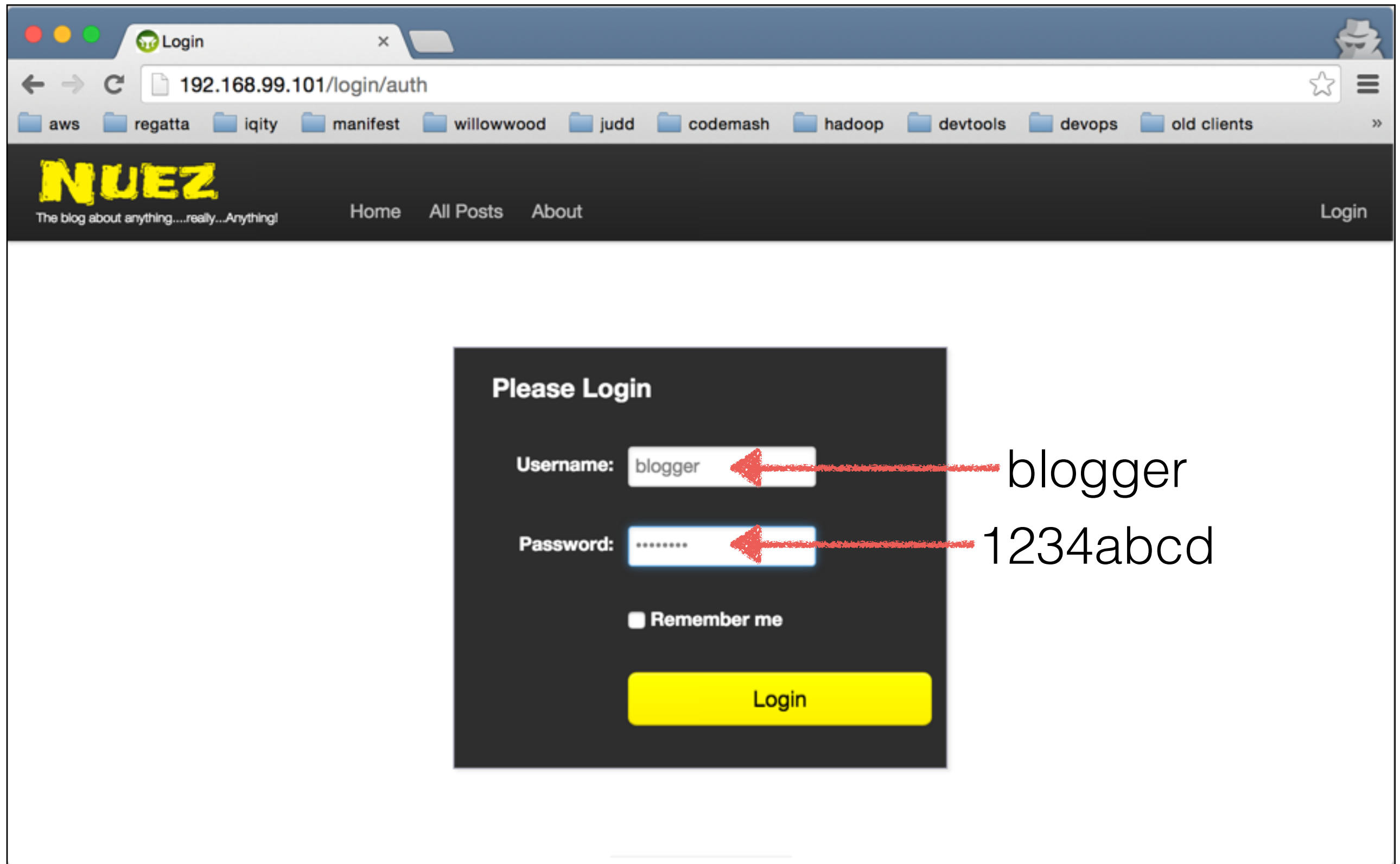
```
docker build -t nuez .
```

```
docker run -d --name nuev --link nuev-db:mysql -p 80:8080 nuev
```

or

```
docker run -it --rm --name nuev --link nuev-db:mysql -p 80:8080 nuev
```





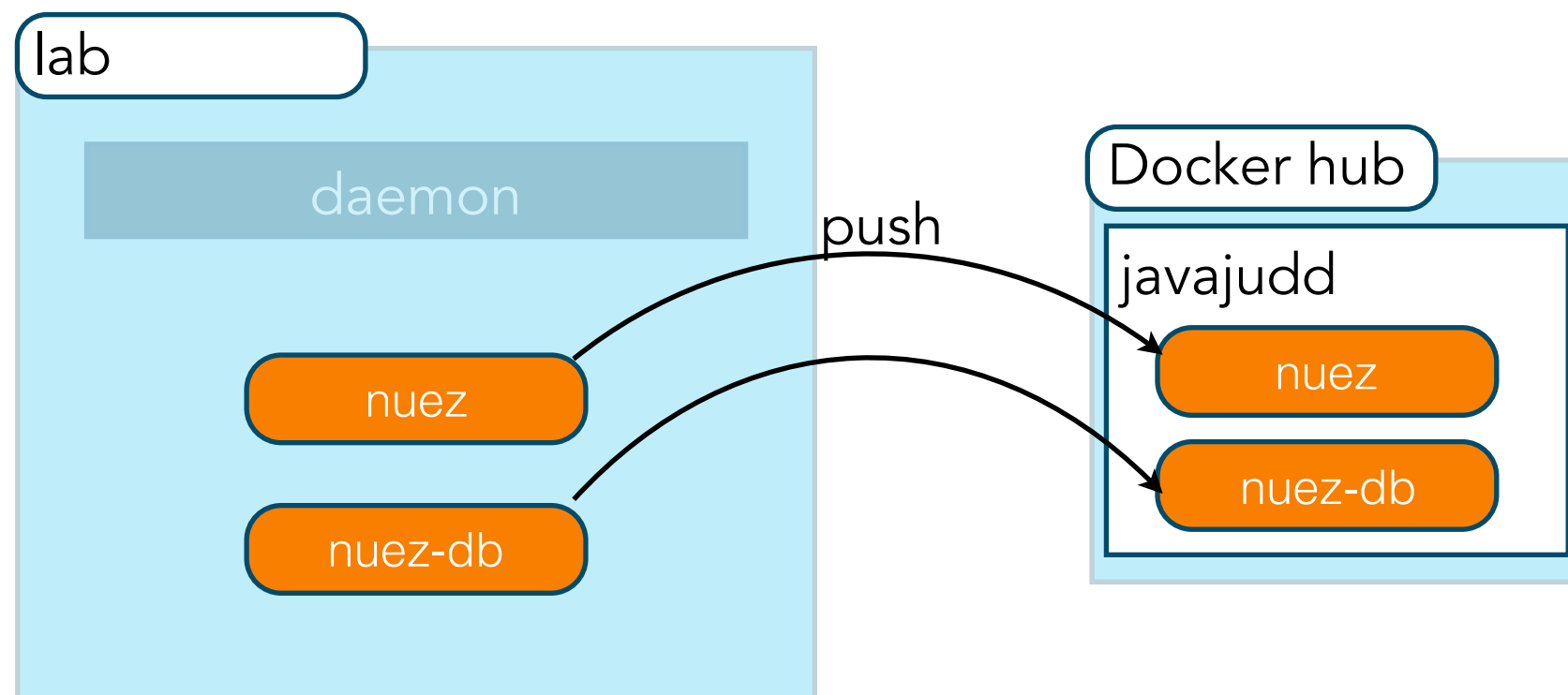
```
docker exec -it nuev bash
```

```
docker diff nuev
```

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



```
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
```



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](#)



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](#)

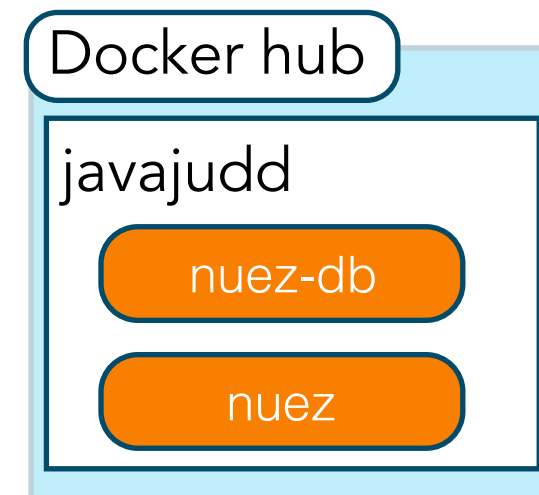
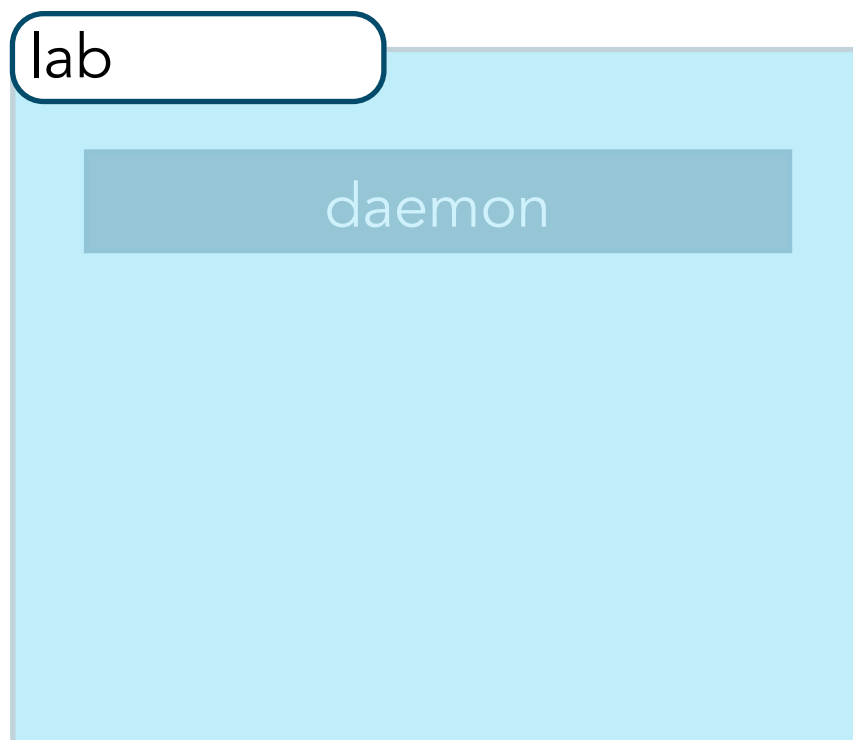
Comments (0)

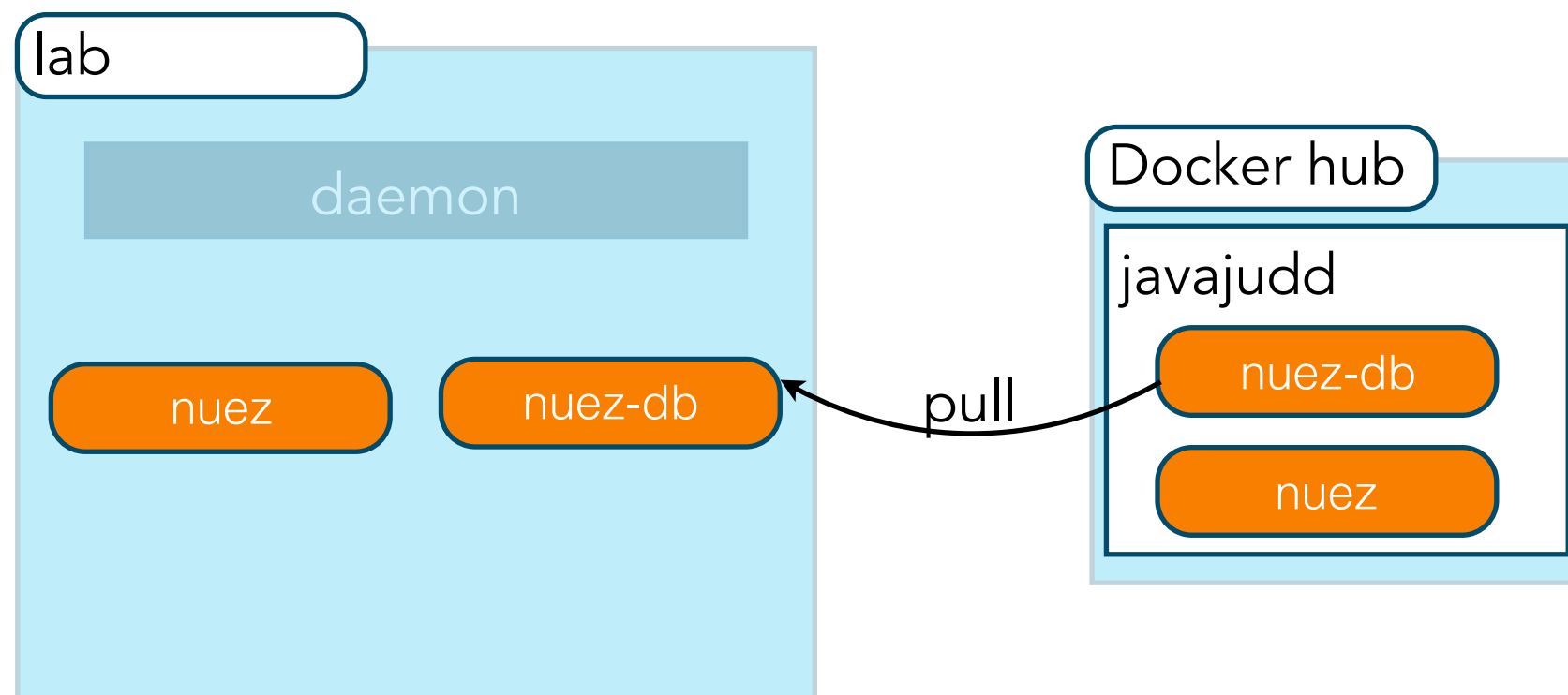
[Add Comment](#)

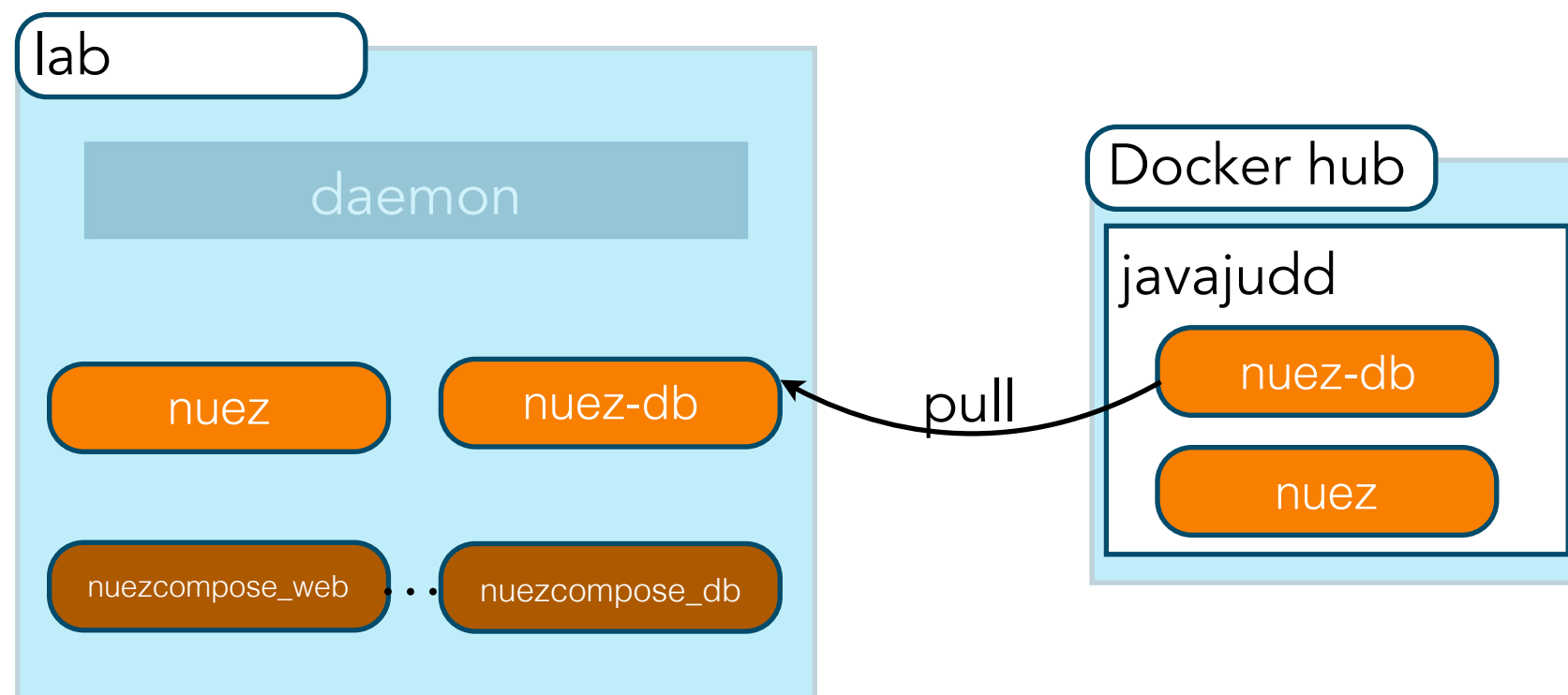
Lab 7

- I. Push nuev and nuev-db docker containers to your repository

COMPOSING







docker-compose.yml

```
web:
  image: javajudd/nuez
  ports:
    - "80:8080"
  links:
    - "db:mysql"
db:
  image: javajudd/nuez-db
  ports:
    - "3306:3306"
  environment:
    MYSQL_USER: nuez-app
    MYSQL_PASSWORD: nuez+1
    MYSQL_DATABASE: nuez
    MYSQL_ROOT_PASSWORD: root+1
```

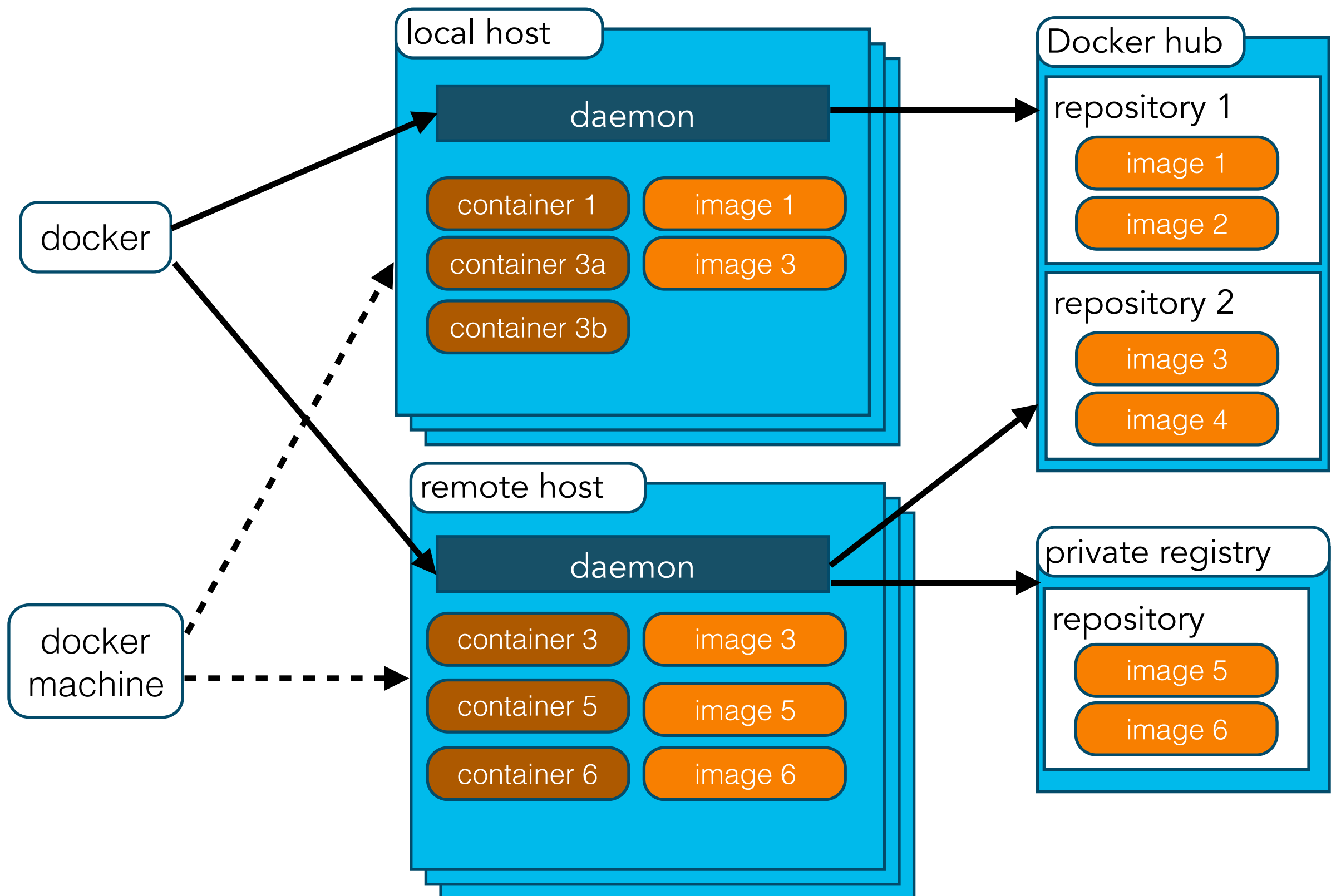
`docker-compose up -d`

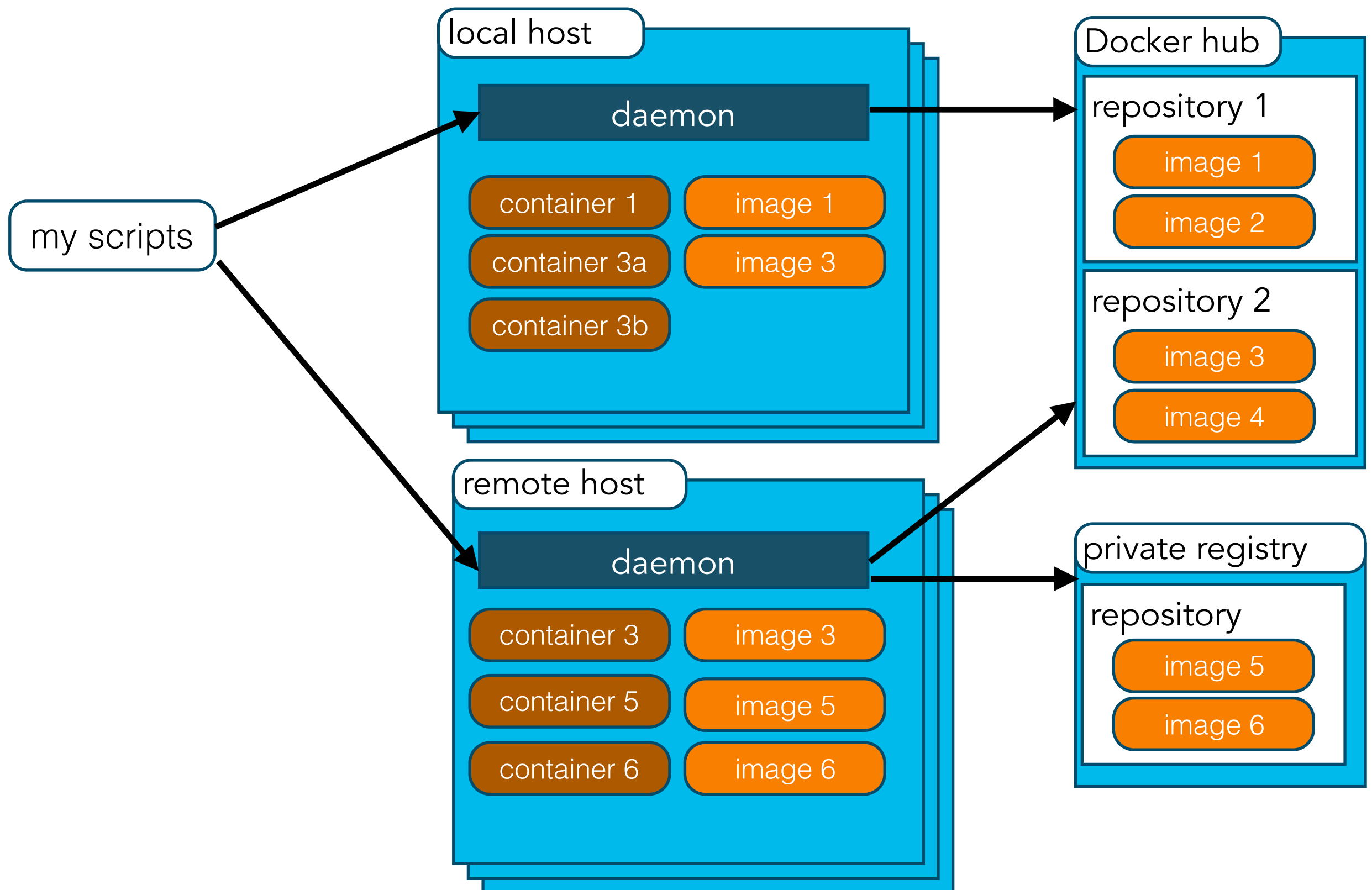
CONTAINER ID	IMAGE	COMMAND	STATUS	PORTS	NAMES
b61f6580e69e13879ea2ae23	javajudd/nuez-db	"/entrypoint.sh mysql"	Up 40 seconds	0.0.0.0:3306->3306/tcp	nuezcompose_db_1

Lab 8

I. Compose nuev and nuev-db

DOCKER API





[Support](#)[Training](#)[Docs](#)[Blog](#)[Docker Hub](#)[Get Started](#)[Products](#)[Customers](#)[Community](#)[Partners](#)[Company](#)[Open Source](#)[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's Remote API uses an open schema model. In this model, unknown properties in incoming messages are ignored. Client applications need to take this behavior into account to ensure they do not break when talking to newer Docker daemons.

The API tends to be REST, but for some complex commands, like attach or pull, the HTTP connection is hijacked to transport STDOUT, STDIN, and STDERR.

By default the Docker daemon listens on `unix:///var/run/docker.sock` and the client must have `root` access to interact with the daemon. If a group named `docker` exists on your system, `docker` applies ownership of the socket to the group.

The current version of the API is v1.21 which means calling `/info` is the same as calling `/v1.21/info`. To call an older version of the API use `/v1.20/info`.

Use the table below to find the API version for a Docker version:

On this page:

[Docker Remote API](#)[Authentication](#)[Using Docker](#)[Machine with the API](#)[Docker Events](#)[Version history](#)[v1.21 API changes](#)[v1.20 API changes](#)[v1.19 API changes](#)[v1.18 API changes](#)[v1.17 API changes](#)[v1.16 API changes](#)[v1.15 API changes](#)[v1.14 API changes](#)

docker-machine ip lab

NAME	ACTIVE	DRIVER	STATE	URL	SWARM
default		virtualbox	Stopped		
lab	*	virtualbox	Running		



```
curl -k --cert $DOCKER_CERT_PATH/cert.pem --key $DOCKER_CERT_PATH/key.pem https://$(docker-machine ip lab):2376/info
```



```
{
  "BridgeNfIp6tables": true,
  "BridgeNfIptables": true,
  "Containers": 4,
  "CpuCfsPeriod": true,
  "CpuCfsQuota": true,
  "Debug": true,
  "DockerRootDir": "/mnt/sda1/var/lib/docker",
  "Driver": "aufs",
  "DriverStatus": [
    [ "Root Dir", "/mnt/sda1/var/lib/docker/aufs" ],
    [ "Backing Filesystem", "extfs" ],
    [ "Dirs", "79" ],
    [ "Dirperm1 Supported", "true" ]
  ],
  "ExecutionDriver": "native-0.2",
  "ExperimentalBuild": false,
  "HttpProxy": "",
  "HttpsProxy": "",
  "ID": "LREF:UAEK:QUM6:ZZV4:5MYN:7EP0:Y5X0:UEHK:IMNU:J7H7:4ZON:FZ5G",
  "IPv4Forwarding": true,
  "Images": 71,
  "IndexServerAddress": "https://index.docker.io/v1/",
  "InitPath": "/usr/local/bin/docker",
  "InitSha1": "",
  "KernelVersion": "4.0.9-boot2docker",
  "Labels": [
    "provider=virtualbox"
  ],
  "LoggingDriver": "json-file",
  "MemTotal": 1044631552,
  "MemoryLimit": true,
  "NCPU": 1,
  "NEventsListener": 0,
  "NFd": 16,
  "NGoroutines": 42,
  "Name": "lab",
  "NoProxy": "",
  "OomKillDisable": true,
  "OperatingSystem": "Boot2Docker 1.8.1 (TCL 6.3); master : 7f12e95 - Thu Aug 13 03:24:56 UTC 2015",
  "RegistryConfig": {
    "IndexConfigs": {
      "docker.io": {
        "Mirrors": null,
        "Name": "docker.io",
        "Official": true,
        "Secure": true
      }
    },
    "InsecureRegistryCIDRs": [
      "127.0.0.0/8"
    ],
    "Mirrors": null
  },
  "SwapLimit": true,
  "SystemTime": "2015-11-12T22:00:05.188478131Z"
}
```

```
openssl pkcs12 -export -inkey $DOCKER_CERT_PATH/key.pem -in $DOCKER_CERT_PATH/cert.pem -name curl-cert  
-out $DOCKER_CERT_PATH/curl-cert.p12 -password pass:mysecret  
curl -k --cert $DOCKER_CERT_PATH/curl-cert.p12:mysecret https://$(docker-machine ip lab):2376/info
```



/containers/json

```
[
  {
    "Command": "/entrypoint.sh mysqld",
    "Created": 1447361102,
    "HostConfig": {
      "NetworkMode": "default"
    },
    "Id": "35c7e85a21af6548370623ee7c42662c1077b93e85b36ab166fea8ac1a83d6d7",
    "Image": "localhost:5000/nuez-db",
    "Labels": {},
    "Names": [
      "/nuez/mysql",
      "/nuez-db"
    ],
    "Ports": [
      {
        "IP": "0.0.0.0",
        "PrivatePort": 3306,
        "PublicPort": 3306,
        "Type": "tcp"
      }
    ],
    "Status": "Up 4 seconds"
  }
]
```

/containers/json?all=1

```
[
  {
    "Command": "catalina.sh run",
    "Created": 1447361114,
    "HostConfig": { "NetworkMode": "default"
    "Id": "cb97f8f0f2e29d0cdb7517a0f9d6d87acbd802e27b8c5b937bec4e41b12dce26",
    "Image": "localhost:5000/nuez",
    "Labels": {},
    "Names": [ "/nuez"
    "Ports": [],
    "Status": "Exited (143) 26 hours ago"
  },
  {
    "Command": "/entrypoint.sh mysqld",
    "Created": 1447361102,
    "HostConfig": { "NetworkMode": "default"
    "Id": "35c7e85a21af6548370623ee7c42662c1077b93e85b36ab166fea8ac1a83d6d7",
    "Image": "localhost:5000/nuez-db",
    "Labels": {},
    "Names": [ "/nuez/mysql", "/nuez-db"
    "Ports": [
      {
        "IP": "0.0.0.0",
        "PrivatePort": 3306,
        "PublicPort": 3306,
        "Type": "tcp"
      }
    ],
    "Status": "Up About a minute"
  },
  {
    "Command": "/bin/registry /etc/docker/registry/config.yml",
    "Created": 1447360363,
    "HostConfig": { "NetworkMode": "default"
    "Id": "063090161f4381b6e776feb0d654e2f4df3c10a3819e4ed1506d023b9729ca35",
    "Image": "registry:2.2.0",
    "Labels": {},
    "Names": [ "/registry"
    "Ports": [],
    "Status": "Exited (2) 26 hours ago"
  }
]
```

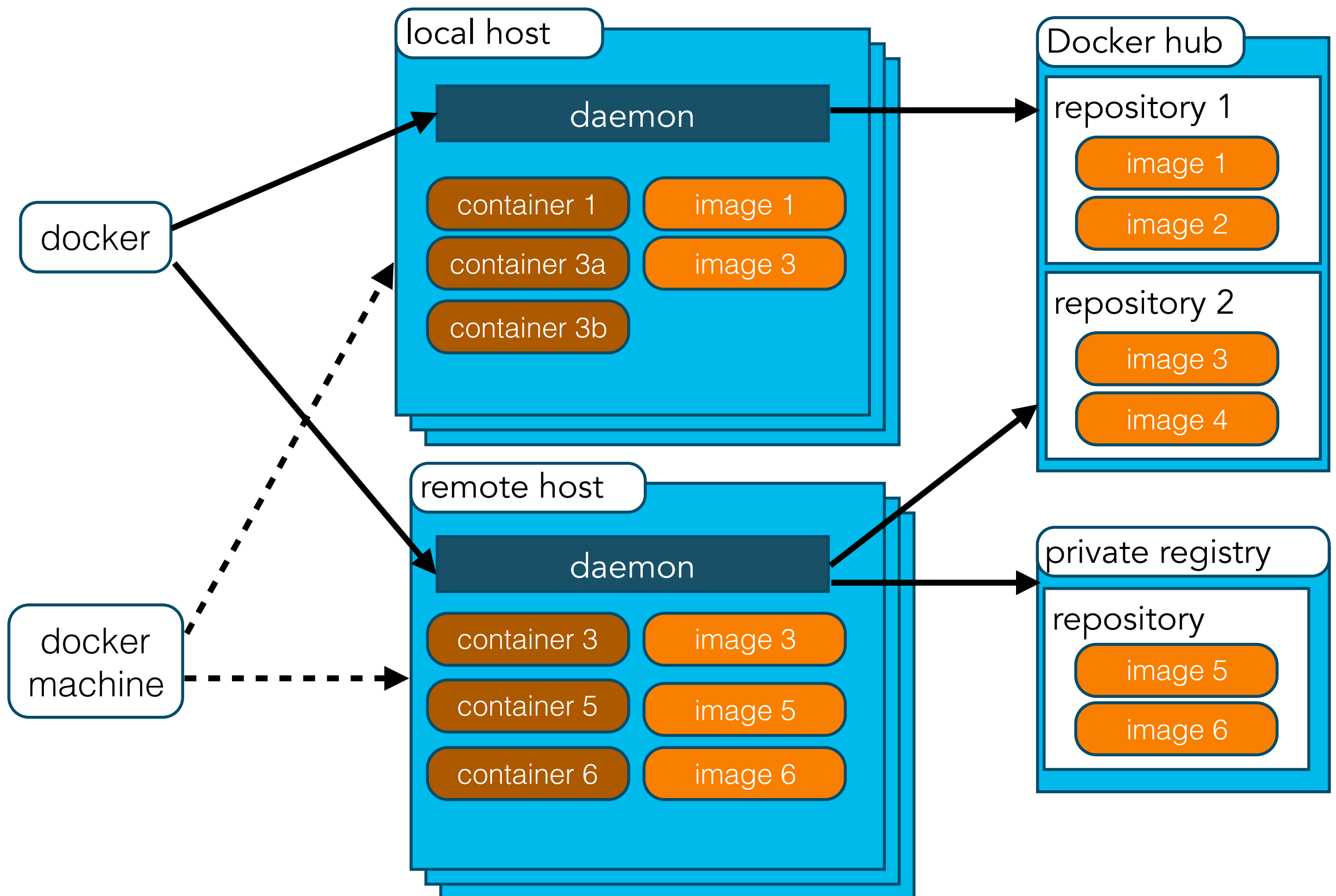
```
curl -k --cert $DOCKER_CERT_PATH/curl-cert.p12:mysecret -X POST  
https://$(docker-machine ip lab):2376/containers/35c7e85a21af/stop
```

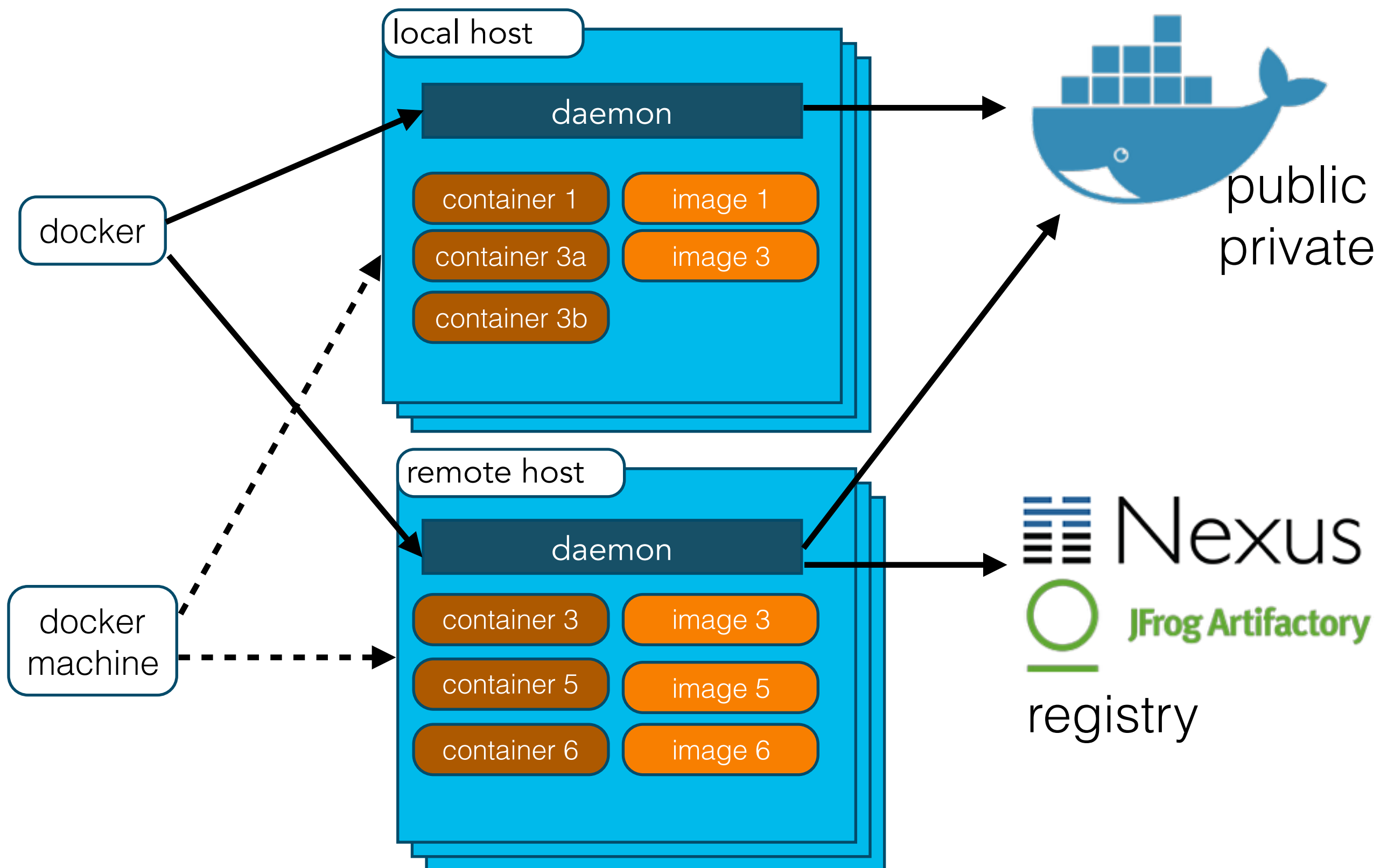
```
curl -k --cert $DOCKER_CERT_PATH/curl-cert.p12:mysecret -X POST  
https://$(docker-machine ip lab):2376/containers/35c7e85a21af/start
```

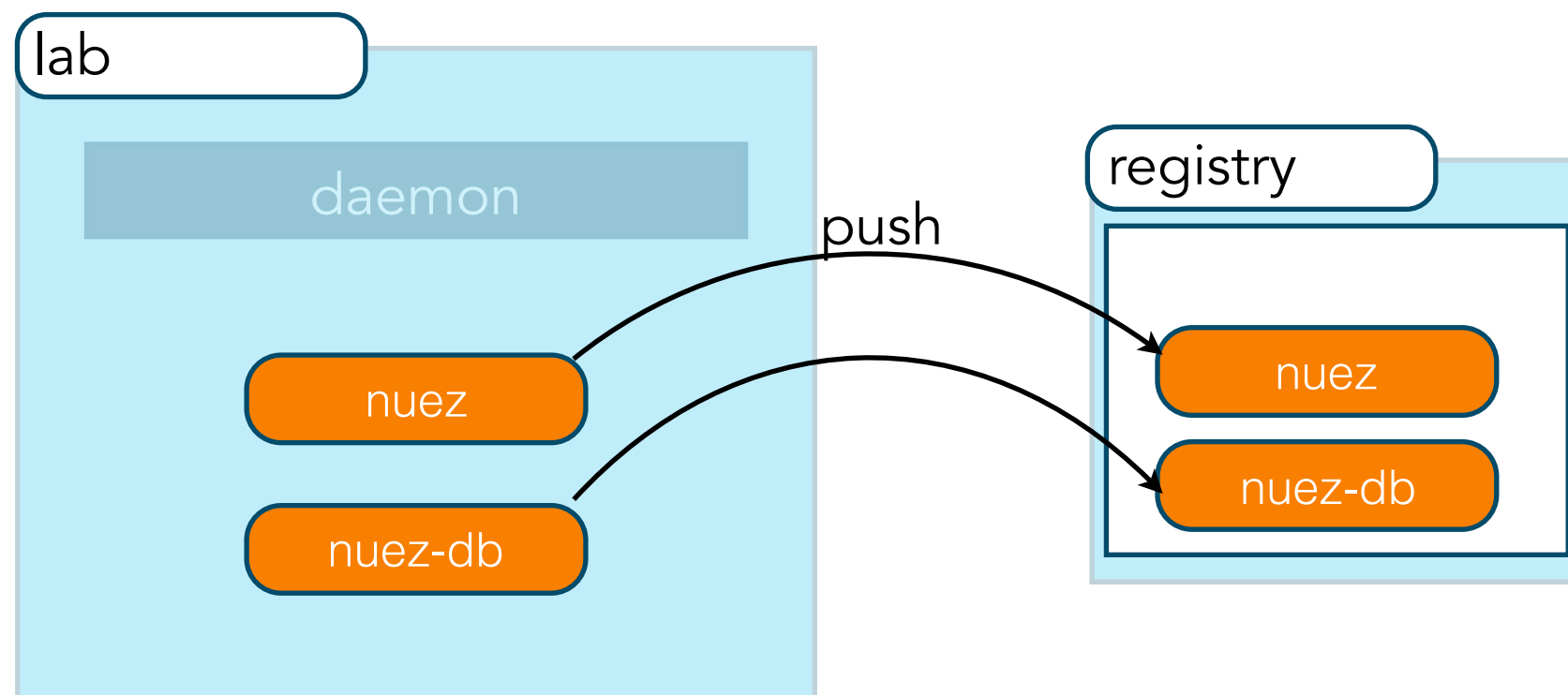
Lab 9

- I. Start and stop the neuz container via the api

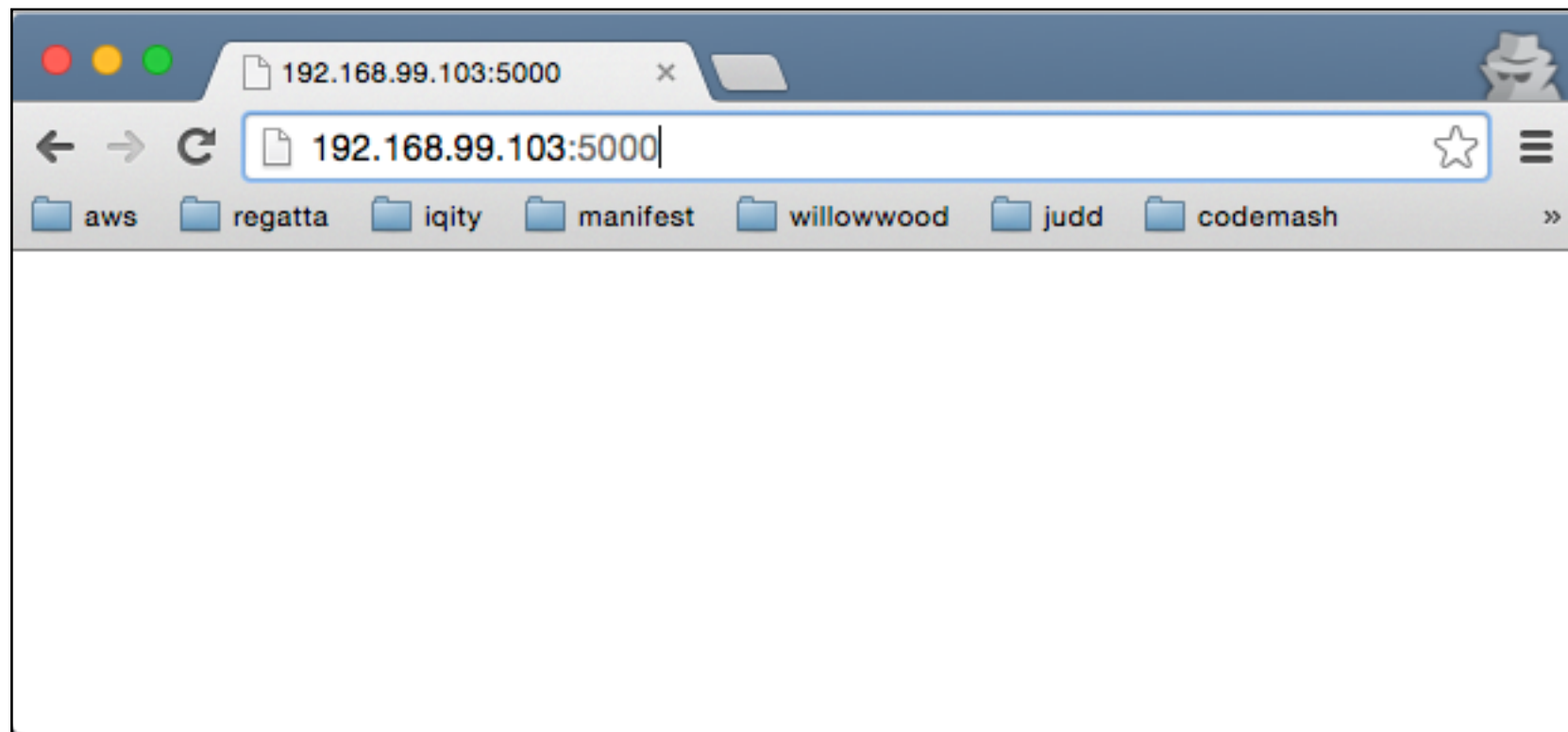
PRIVATE REGISTRIES







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



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

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

[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.

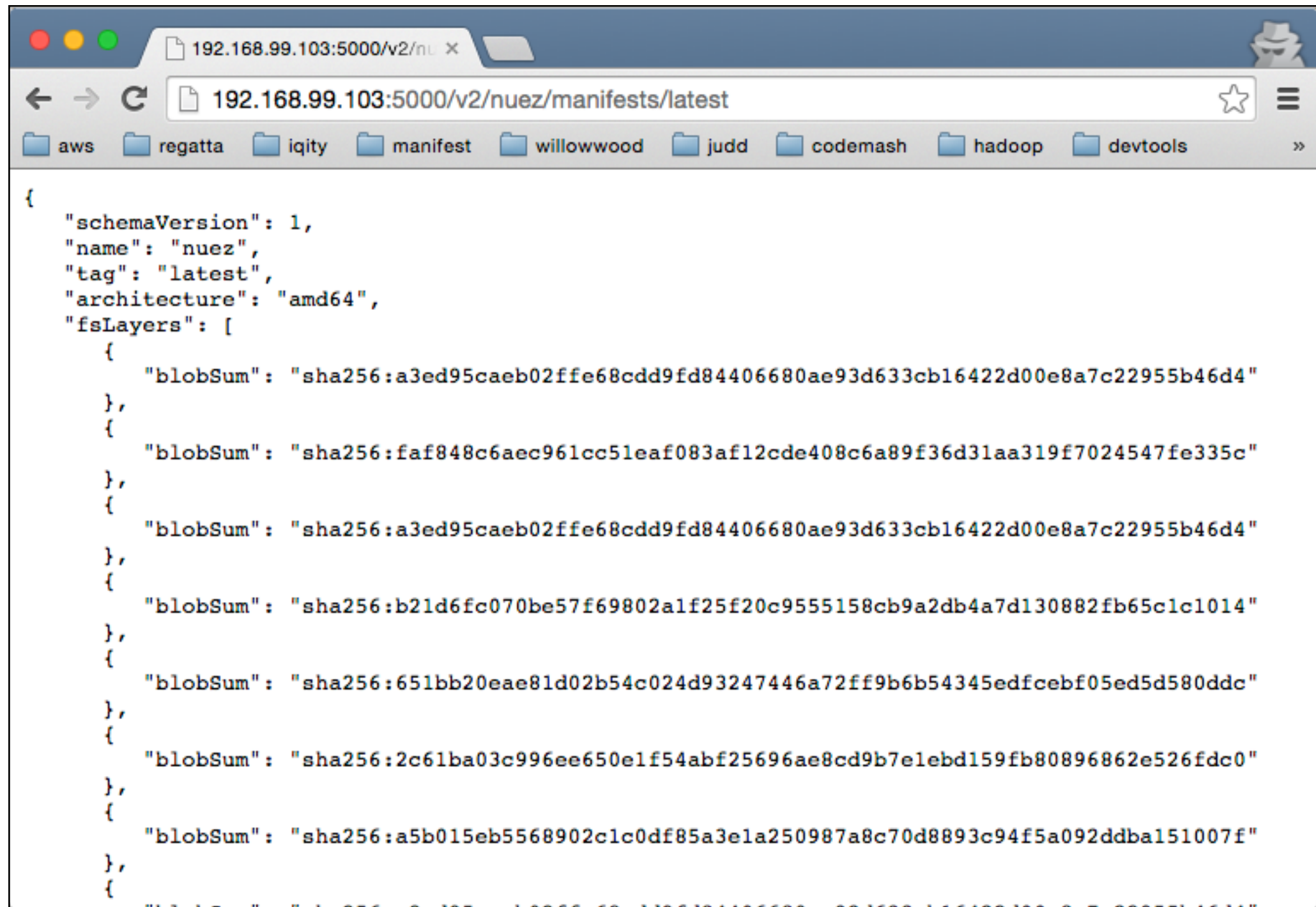
For relevant details and history leading up to this specification, please see the following issues:

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>

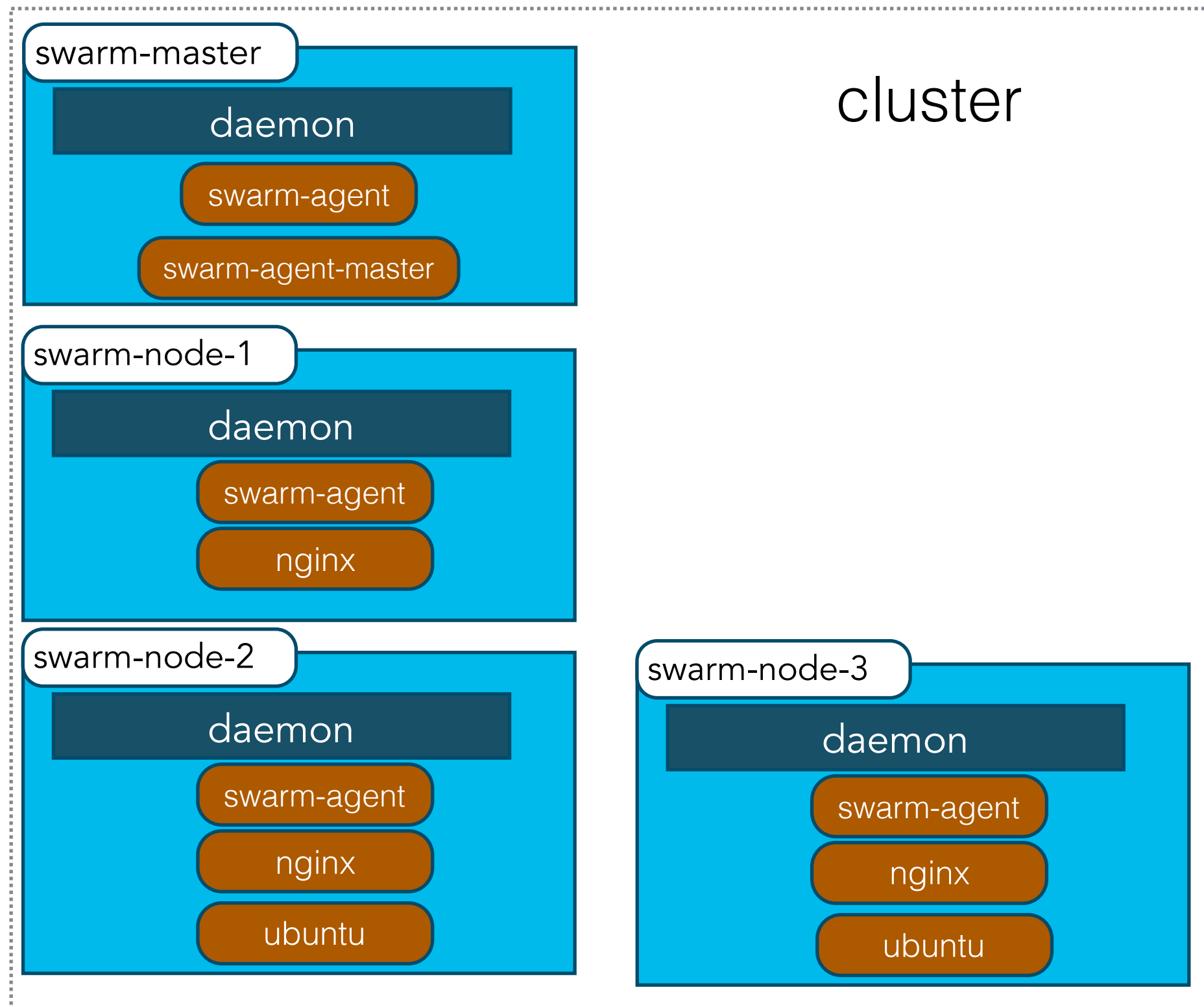
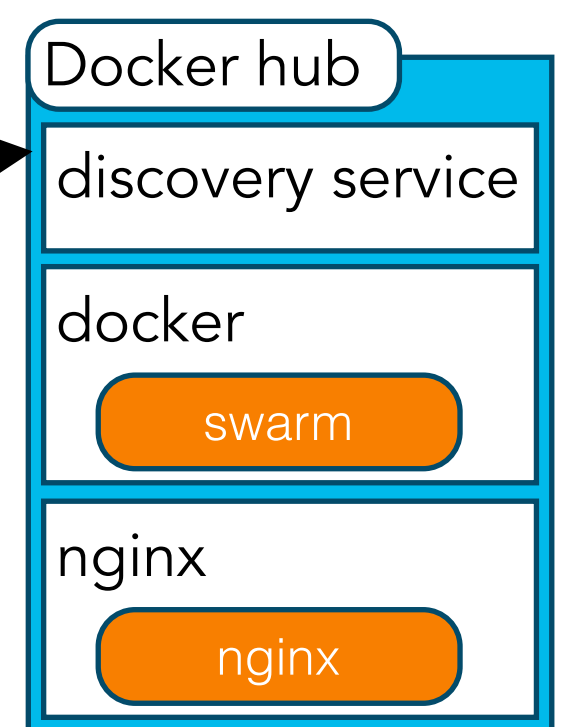


A screenshot of a web browser window displaying a JSON manifest. The browser's address bar shows the URL `192.168.99.103:5000/v2/nuez/manifests/latest`. Below the address bar, there is a navigation bar with several folder icons labeled `aws`, `regatta`, `iqity`, `manifest`, `willowwood`, `judd`, `codemash`, `hadoop`, and `devtools`. The main content area displays a JSON object representing a manifest. The JSON is formatted with indentation and includes the following fields: `schemaVersion` (1), `name` ("nuez"), `tag` ("latest"), `architecture` ("amd64"), and `fsLayers` (an array of six objects, each containing a `blobSum` field with a SHA256 hash). The browser interface includes standard macOS window controls (red, yellow, green buttons) in the top-left corner and a search icon in the top-right corner.

```
{
  "schemaVersion": 1,
  "name": "nuez",
  "tag": "latest",
  "architecture": "amd64",
  "fsLayers": [
    {
      "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4"
    },
    {
      "blobSum": "sha256:faf848c6aec961cc51eaf083af12cde408c6a89f36d31aa319f7024547fe335c"
    },
    {
      "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4"
    },
    {
      "blobSum": "sha256:b21d6fc070be57f69802a1f25f20c9555158cb9a2db4a7d130882fb65c1c1014"
    },
    {
      "blobSum": "sha256:651bb20eae81d02b54c024d93247446a72ff9b6b54345edfceb05ed5d580ddc"
    },
    {
      "blobSum": "sha256:2c61ba03c996ee650e1f54abf25696ae8cd9b7e1ebd159fb80896862e526fdc0"
    },
    {
      "blobSum": "sha256:a5b015eb5568902c1c0df85a3e1a250987a8c70d8893c94f5a092ddba151007f"
    },
    {
      "blobSum": "sha256:..."
    }
  ]
}
```

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

SWARM



```
docker pull swarm  
sid=$(docker run swarm create)  
echo $sid
```

```
b00ee410d9d156eac06cac4176047563
```

```
docker-machine create -d virtualbox --swarm --swarm-master --swarm-discovery token://$sid swarm-master
```

```
docker-machine create -d virtualbox --engine-label itype=frontend --swarm --swarm-discovery token://$sid swarm-node-01
```

```
docker-machine create -d virtualbox --swarm --swarm-discovery token://$sid swarm-node-02
```

```
docker-machine create -d virtualbox --swarm --swarm-discovery token://$sid swarm-node-03
```

```
eval "$(docker-machine env --swarm swarm-master)"
```



docker info

```
Containers: 6
Images: 4
Role: primary
Strategy: spread
Filters: health, port, dependency, affinity, constraint
Nodes: 4
  swarm-master: 192.168.99.108:2376
    L Containers: 2
    L Reserved CPUs: 0 / 1
    L Reserved Memory: 0 B / 1.022 GiB
    L Labels: executiondriver=native-0.2, kernelversion=4.0.9-boot2docker, operatingsystem=Boot2Docker
1.8.1 (TCL 6.3); master : 7f12e95 - Thu Aug 13 03:24:56 UTC 2015, provider=virtualbox, storagedriver=aufs
  swarm-node-01: 192.168.99.109:2376
    L Containers: 1
    L Reserved CPUs: 0 / 1
    L Reserved Memory: 0 B / 1.022 GiB
    L Labels: executiondriver=native-0.2, itype=frontend, kernelversion=4.0.9-boot2docker,
operatingsystem=Boot2Docker 1.8.1 (TCL 6.3); master : 7f12e95 - Thu Aug 13 03:24:56 UTC 2015,
provider=virtualbox, storagedriver=aufs
  swarm-node-02: 192.168.99.110:2376
    L Containers: 1
    L Reserved CPUs: 0 / 1
    L Reserved Memory: 0 B / 1.022 GiB
    L Labels: executiondriver=native-0.2, kernelversion=4.0.9-boot2docker, operatingsystem=Boot2Docker
1.8.1 (TCL 6.3); master : 7f12e95 - Thu Aug 13 03:24:56 UTC 2015, provider=virtualbox, storagedriver=aufs
  swarm-node-03: 192.168.99.111:2376
    L Containers: 2
    L Reserved CPUs: 0 / 1
    L Reserved Memory: 0 B / 1.022 GiB
    L Labels: executiondriver=native-0.2, kernelversion=4.0.9-boot2docker, operatingsystem=Boot2Docker
1.8.1 (TCL 6.3); master : 7f12e95 - Thu Aug 13 03:24:56 UTC 2015, provider=virtualbox, storagedriver=aufs
CPUs: 4
Total Memory: 4.086 GiB
Name: d0caecd5c477
```

```
docker run -d -p 80:80 --name nginx1 nginx
for i in `seq 1 6`; do docker run -itd -e constraint:itype!=frontend --name eng$i
ubuntu; done
```

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



Log In

Log In

[Forgot password?](#)

Don't have an account? [Sign Up](#)

[Droplets](#)[Images](#)[Networking](#)[API](#)[Support](#)[Create Droplet](#)

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



77847e8e7d07c02135767096fdaaf8e5

[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
```

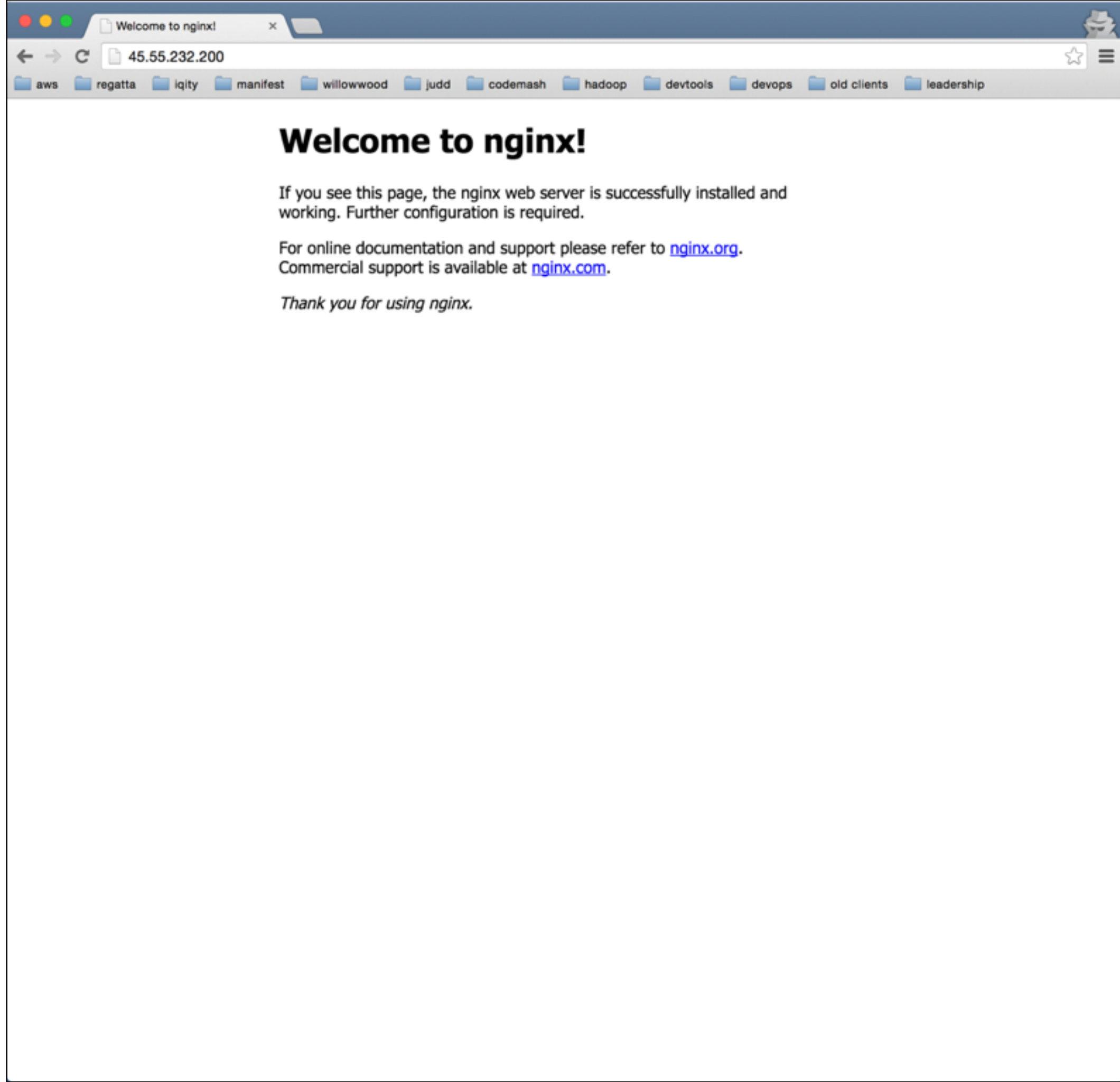
Creating SSH key...

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!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

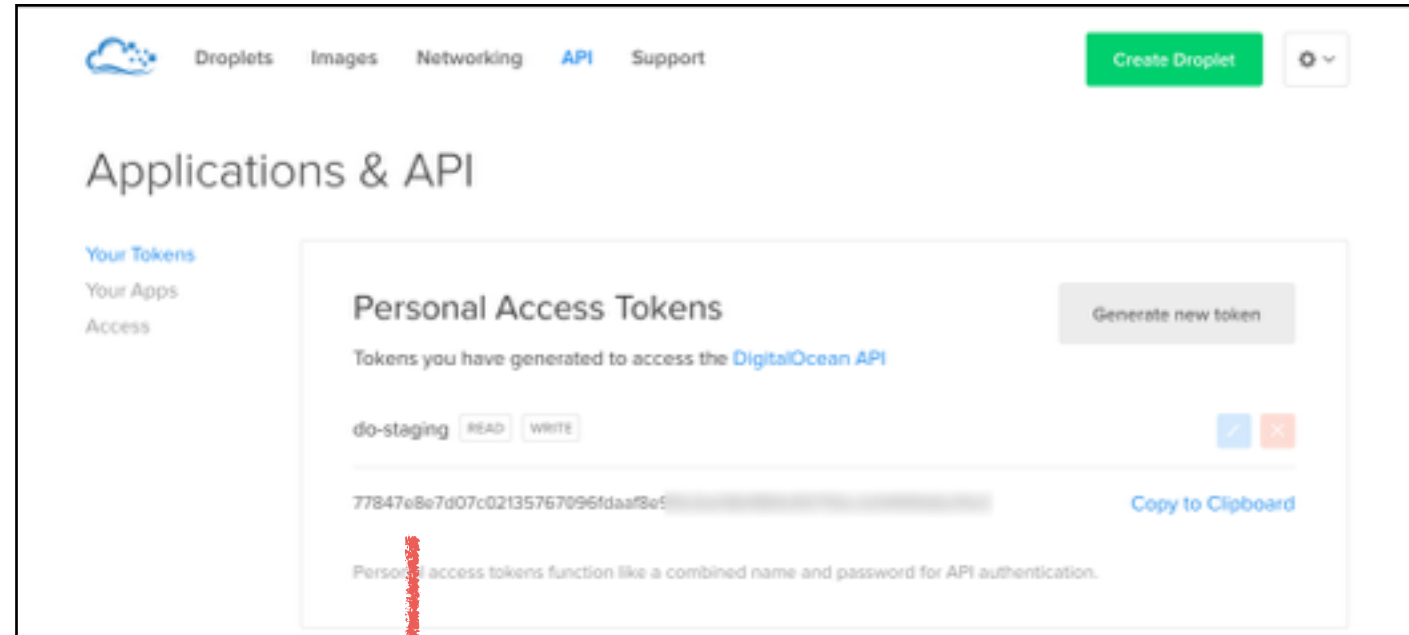
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

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



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](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 ▾](#)

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](#)[IAM](#) > [Users](#) > [redacted]

◄ 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](#)[IAM](#) > [Users](#) > [redacted]

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][Manage Password](#)**Password** Yes**Last Used** 2015-10-23 08:59 EDT

[Dashboard](#)[Details](#)[Groups](#)[Users](#)[Roles](#)[Policies](#)[Identity Providers](#)[Account Settings](#)[Credential Report](#)[Encryption Keys](#)[IAM](#) > [Users](#) > [redacted]

Summary

User ARN: arn:aws:iam::[redacted]:user/[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](#)

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](#)[IAM](#) > [Users](#) > [redacted]

Summary

User ARN: arn:aws:iam::[redacted]:user/[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.

▼ [Hide User Security Credentials](#)



Access Key ID: AKIAIN[redacted]
Secret Access Key: Dbig4+636-[redacted]

[Close](#)[Download Credentials](#)

Last Used 2015-10-23 08:59 EDT



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 properties X

<< < 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

State: available

VPC CIDR: 10.0.0.0/16

DHCP options set: dopt-70829f12

Network ACL: acl-dadd29be

Tenancy: Default

DNS resolution: yes

DNS hostnames: yes



```
docker-machine -D create \  
  --driver amazec2 \  
  --amazec2-access-key AKIAIMN \  
  --amazec2-secret-key Dbig4+6364 \  
  --amazec2-vpc-id vpc-c602eaa2 \  
  --amazec2-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



Feedback

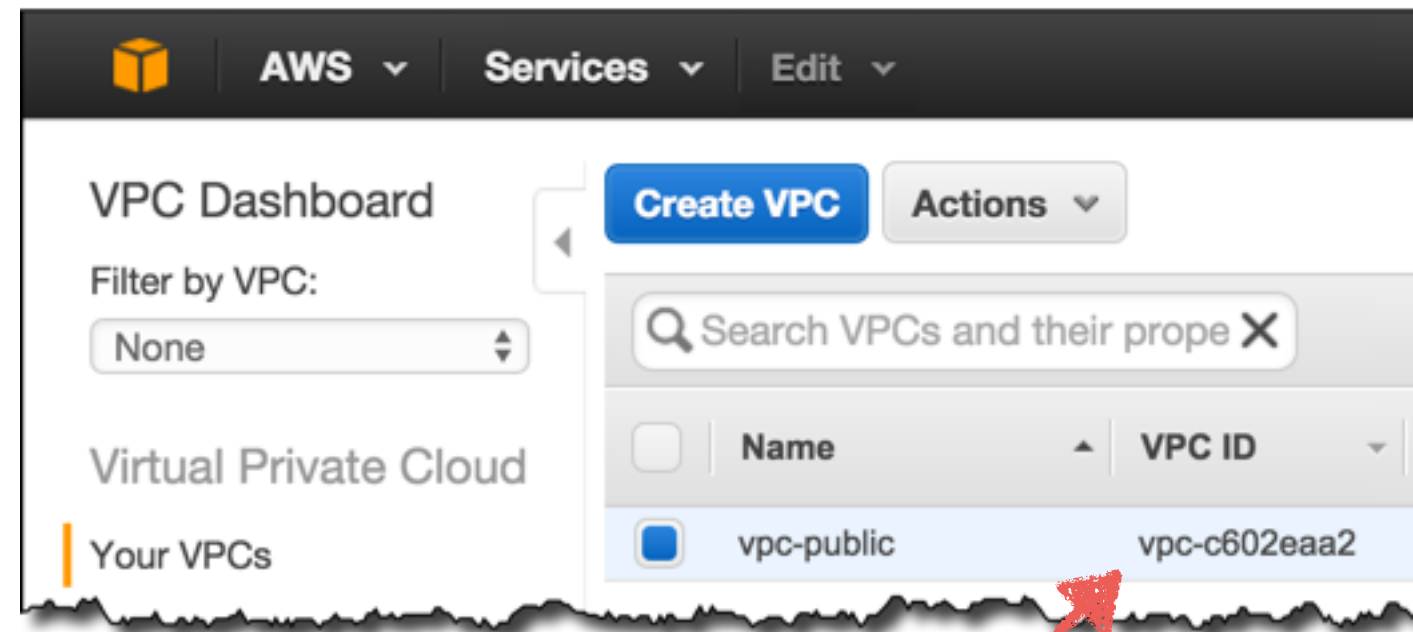
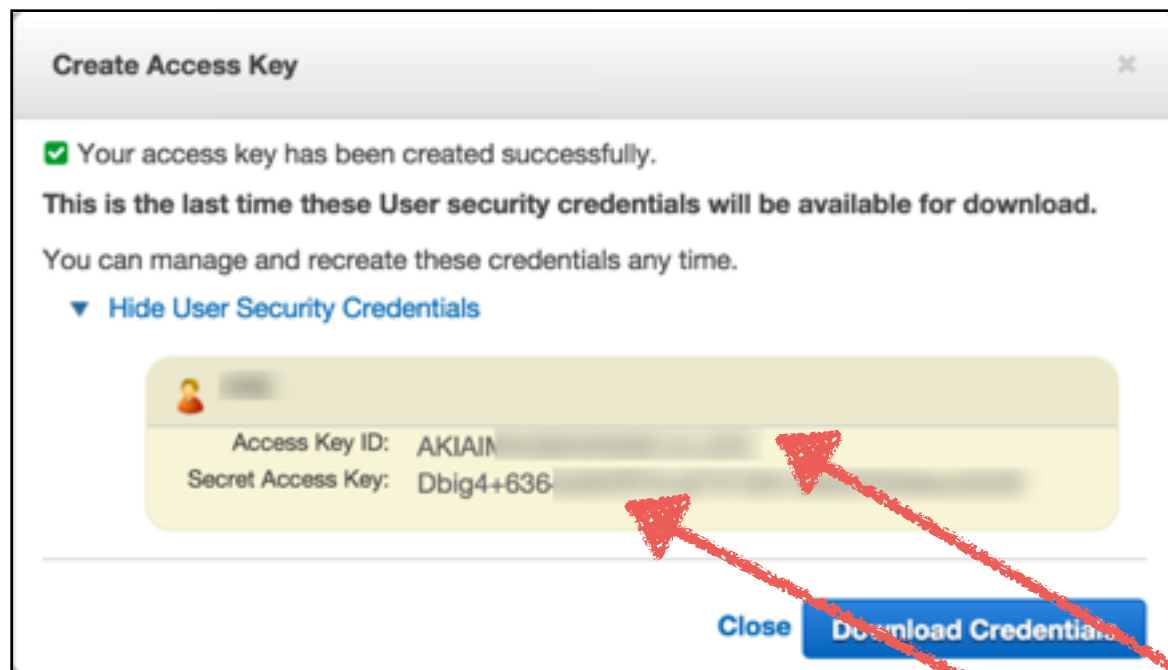


English

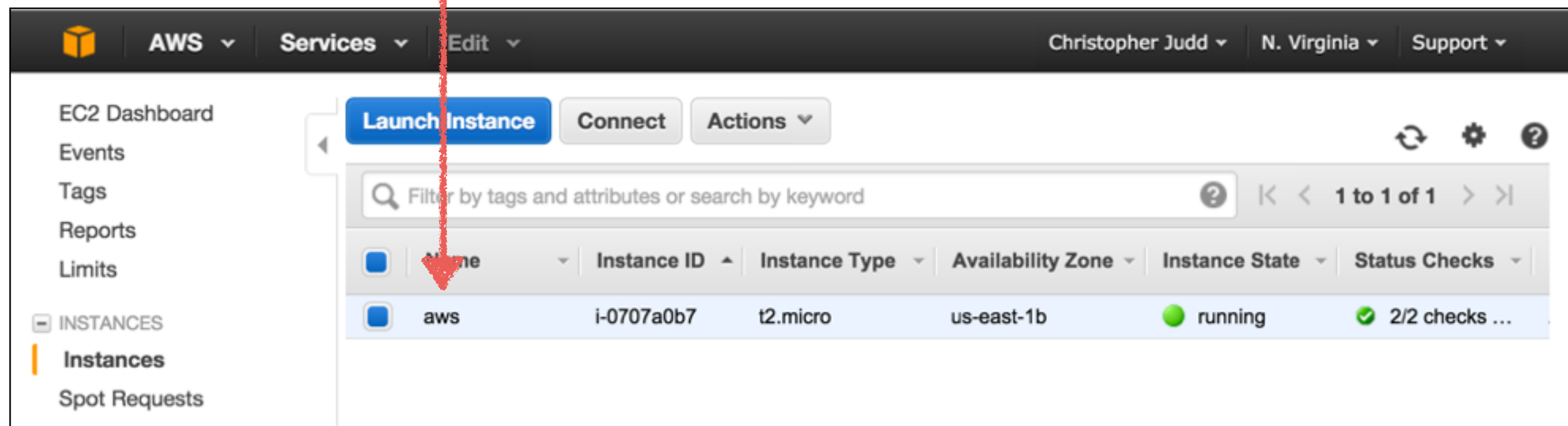
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Privacy Policy

Terms of Use



```
docker-machine -D create \
  --driver amazec2 \
  --amazec2-access-key AKIAIMN \
  --amazec2-secret-key Dbig4+6364 \
  --amazec2-vpc-id vpc-c602eaa2 \
  --amazec2-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 ▾



sg-d477e2b2

docker-machine

vpc-c602eaa2

**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 In

Commands

IMAGES

AMIs

Bundle Tas

ELASTIC BL

Volumes

Snapshots

NETWORK &

Security G

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

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

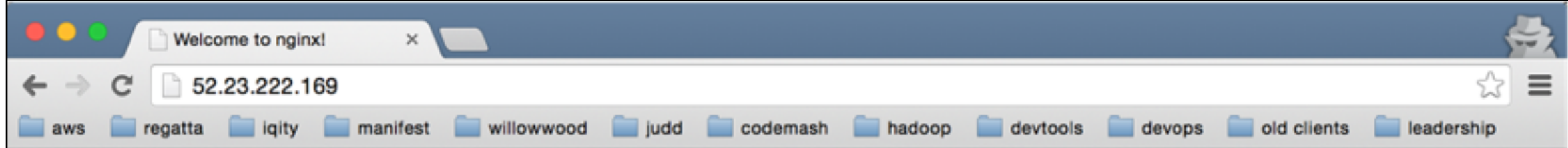
0.0.0.0/0

Custom TCP Rule

TCP

2376

0.0.0.0/0



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

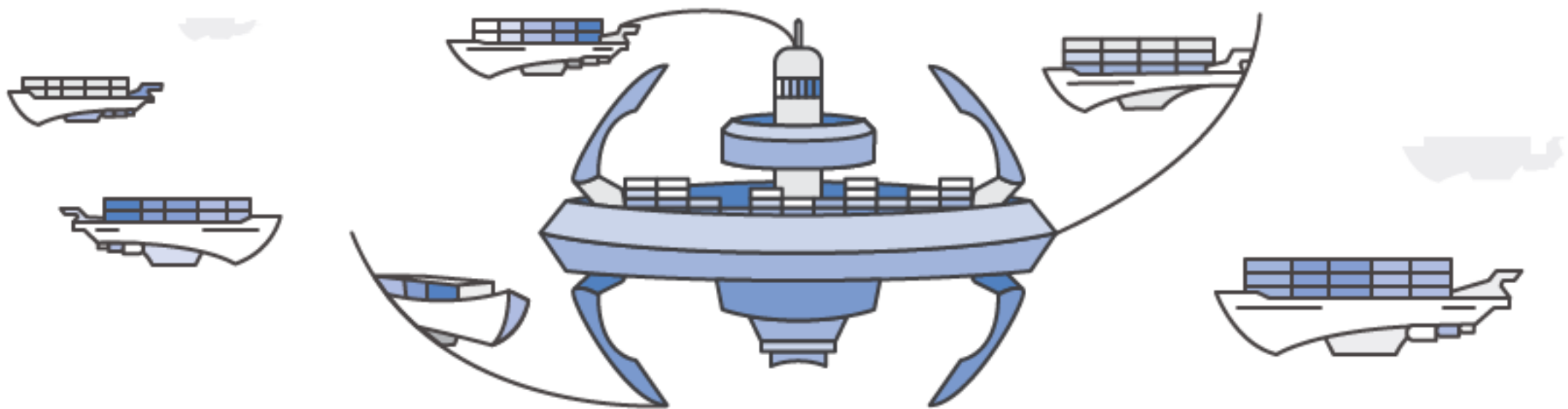
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

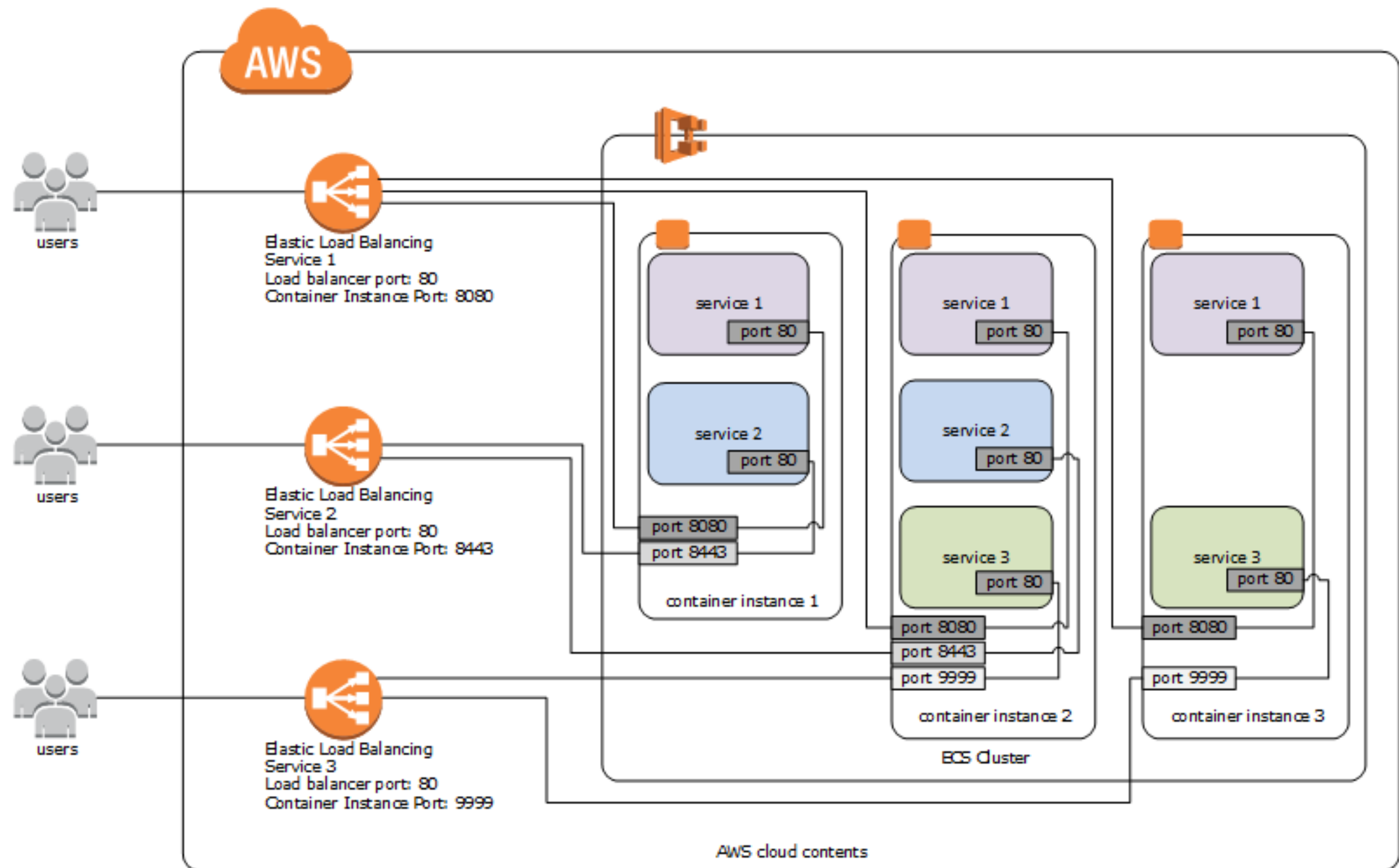
Thank you for using nginx.

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

EC2 Container Service







Amazon EC2 Container Service (ECS)



Amazon ECS makes it easy to deploy, manage, and scale Docker containers running applications, services, and batch processes. Amazon ECS places containers across your cluster based on your resource needs and is integrated with familiar features like Elastic Load Balancing, EC2 security group, EBS volumes and IAM roles.

[Get started](#)

[Learn more about Amazon ECS](#)



Run containers at scale

Amazon ECS makes it easy to use containers a



Flexible container placement

Amazon ECS lets you schedule long-running



Integrated and extensible

Amazon ECS is integrated with familiar features lik



Getting Started with Amazon EC2 Container Service

Step 1: Welcome to Amazon ECS

Step 2: Create Task Definition

Step 3: Schedule Tasks

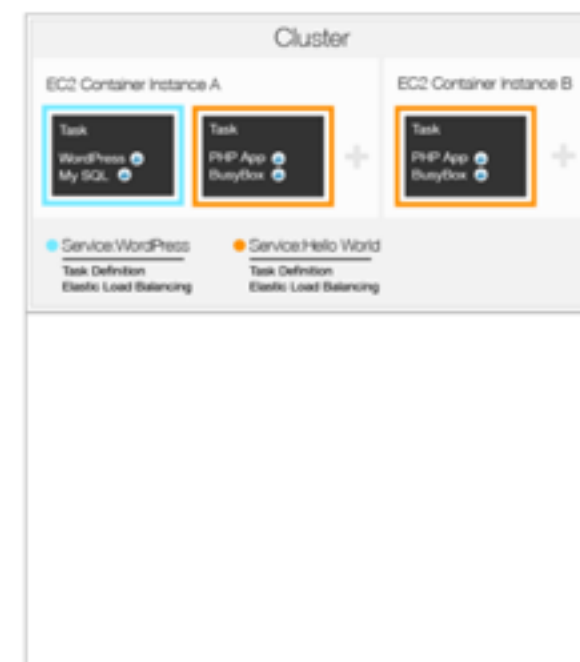
Step 4: Configure Cluster

Step 5: Review

Welcome to Amazon ECS

This wizard will guide you through the process to get started with Amazon ECS. You will:

- **Create a task definition**
Describe the components of your application such as the Docker containers to run, the resources they will use, and how they link together.
- **Schedule tasks**
Reference the number of task definitions to run as tasks or start a service that can use Elastic Load Balancing to route traffic to the containers.
- **Configure the cluster**
Set options for the logical grouping of **container** instances used to run the tasks.



To get started, select a sample task definition or create a custom task definition:

- ☐ Amazon ECS sample
Create a task definition using amazon-ecs-sample image.
- ☒ Custom
Register a custom task definition.

Cancel

Next Step



Getting Started with Amazon EC2 Container Service

[Step 1: Welcome to Amazon ECS](#)

Step 2: Create custom Task Definition

[Step 3: Schedule Tasks](#)

[Step 4: Configure Cluster](#)

[Step 5: Review](#)

Create custom Task Definition

You can modify the parameters in the task definition (for example, to provide more CPU resources or change the port mappings) to suit your particular application. For more information, see Task Definitions in the [Amazon ECS Developer Guide](#).

Task Definition Name*

nginx



Container Definitions



Container Name	Image	Memory (MB)	Essential	
----------------	-------	-------------	-----------	--

[+ Add Container Definition](#)

Volumes



Name	Source Path	
------	-------------	--

[+ Add volume](#)

[Configure via JSON](#)

* Required

[Cancel](#)

[Previous](#)

[Next Step](#)



AWS ▾

Services ▾

Edit ▾

Getting Started with Amazon ECS

[Step 1: Welcome to Amazon ECS](#)**Step 2: Create custom Task Definition**[Step 3: Schedule Tasks](#)[Step 4: Configure Cluster](#)[Step 5: Review](#)

Add container



STANDARD

Container name*

nginx-cn



Image*

nginx



Memory (MB)*

512



Port mappings

Host port

Container port

Protocol



80

80

tcp ▾

[+ Add port mapping](#)[▶ Advanced container configuration](#)

* Required

[Cancel](#)[Add](#)



Getting Started with Amazon EC2 Container Service

[Step 1: Welcome to Amazon ECS](#)**Step 2: Create custom Task Definition**[Step 3: Schedule Tasks](#)[Step 4: Configure Cluster](#)[Step 5: Review](#)

Create custom Task Definition

You can modify the parameters in the task definition (for example, to provide more CPU resources or change the port mappings) to suit your particular application. For more information, see Task Definitions in the [Amazon ECS Developer Guide](#).

Task Definition Name*



Container Definitions

Container Name	Image	Memory (MB)	Essential	
nginx-cn	nginx	512	true	

[+ Add Container Definition](#)

Volumes

Name	Source Path	
------	-------------	--

[+ Add volume](#)[Configure via JSON](#)

* Required

[Cancel](#)[Previous](#)[Next Step](#)



Getting Started with Amazon EC2 Container Service

[Step 1: Welcome to Amazon ECS](#)[Step 2: Create custom Task Definition](#)**Step 3: Schedule Tasks**[Step 4: Configure Cluster](#)[Step 5: Review](#)

Schedule Tasks

A task is an instantiation of the task definition created in step 2 that runs on a container instance.

**Run Tasks Once**

You can run individual tasks for processes such as batch jobs that perform work and then stop.

**Create a service**

You can create a service for long running applications. A service auto recovers stopped tasks to maintain the desired number of tasks and can optionally register tasks with one or more ELBs. You can update the service to deploy a new image or change the running number of tasks

Desired number of Tasks*

1

Service Name*

nginx-webapp

Elastic Load Balancing

Container: Port*

nginx-cn:80

ELB protocol*

HTTP ▾

ELB port*

80

Ping port

80

Ping protocol*

HTTP ▾

Ping path*

/

* Required

[Cancel](#)[Previous](#)[Next Step](#)



Getting Started with Amazon EC2 Container Service

[Step 1: Welcome to Amazon ECS](#)[Step 2: Create custom Task Definition](#)[Step 3: Schedule Tasks](#)**Step 4: Configure Cluster**[Step 5: Review](#)

Configure Cluster

Before you can run tasks in your Amazon ECS cluster, you must launch container instances into it. An Amazon ECS container instance is an Amazon EC2 instance with the Amazon ECS Container Agent running on it. The agent makes calls on your behalf by using an IAM role configured for Amazon ECS.

Cluster* default

Number of Instances*

1

AMI*

[Amazon ECS-Optimized Amazon Linux AMI \(ami-d74357b6\)](#)

Instance Type*

t2.micro

Key pair name*

None



You will not be able to SSH into your instances without selecting a key pair.

Security Group

Your instances may be accessible from any IP address. We recommend that you update the below security group ingress rule to allow access from known IP addresses only. ECS automatically opens up port 80 to facilitate access to the application or service you're running. Additionally, we open port 80 for Elastic Load Balancer access.

Security Ingress CIDR*

0.0.0.0/0

IAM Role Information

The IAM role for EC2 refers to an IAM role that grants your EC2 instances permissions to access AWS resources it needs, including the ability to connect with the ECS service to manage your tasks. The IAM role for ECS Service grants the necessary permissions for ECS to interact with resources such as Elastic Load Balancing. Clicking on the button below will open a new tab to the IAM role creation one click wizard

ECS instance role*

ecsInstanceRole

or

ECS service role*

ecsServiceRole

[Create IAM Roles](#)

* Required

[Cancel](#)[Previous](#)[Review & Launch](#)



Getting Started with Amazon EC2 Container Service

[Step 1: Welcome to Amazon ECS](#)[Step 2: Create custom Task Definition](#)[Step 3: Schedule Tasks](#)[Step 4: Configure Cluster](#)[Step 5: Review](#)

Review

Review your task definition, service and container instance details.

Task Definition

[Edit](#)

Task Definition Name nginx

Task Definition

```
{
  "family": "nginx",
  "volumes": [],
  "containerDefinitions": [
    {
      "name": "nginx-cn",
      "image": "nginx",
      "memory": "512",
      "essential": true,
      "portMappings": [
        {
          "hostPort": "80",
          "containerPort": "80",
          "protocol": "tcp"
        }
      ]
    }
  ]
}
```

Task Configurations

[Edit](#)

Service Name nginx-webapp

Number of Tasks 1

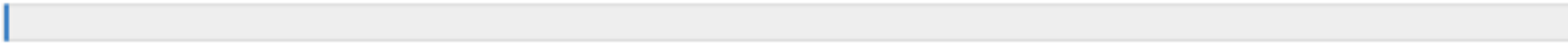
Container: Host Port nginx-cn:80



Launch status

Your container instances are launching, and it may take a few minutes until they are in the running state, and ready to access. Usage hours on your new container instances will start immediately and continue to accrue until you stop or terminate.

ECS Instances status



Launching the requested resources using AWS CloudFormation. This may take a few minutes. [Click here to view the AWS::CloudFormation::Stack... CloudFormation stack.](#)

ECS Status



Waiting for ECS instance to launch...

This may take a few minutes

[View Service](#)



Create Stack

Actions

Design template



Filter: Active

By Name:

Showing 2 stacks

	Stack Name	Created Time	Status	Description
☒	EC2ContainerService-default-1	2015-11-10 18:57:12 UTC-0500	CREATE_IN_PROGRESS	AWS CloudFormation template to create a new VPC or use an
☐	awseb-e-zmgdavufct-stack	2015-10-28 13:49:24 UTC-0400	UPDATE_COMPLETE	AWS Elastic Beanstalk environment (Name: 'medone-env' Id:

Overview

Outputs

Resources

Events

Template

Parameters

Tags

Stack Policy



2015-11-10	Status	Type	Logical ID	Status Reason
18:58:25 UTC-0500	CREATE_IN_PROGRESS	AWS::AutoScaling::AutoScalingGroup	EcsInstanceAsg	
▶ 18:58:23 UTC-0500	CREATE_COMPLETE	AWS::AutoScaling::LaunchConfiguration	EcsInstanceLcWithoutKeyPair	
▶ 18:58:22 UTC-0500	CREATE_IN_PROGRESS	AWS::AutoScaling::LaunchConfiguration	EcsInstanceLcWithoutKeyPair	Resource creation Initiated
18:58:22 UTC-0500	CREATE_IN_PROGRESS	AWS::AutoScaling::LaunchConfiguration	EcsInstanceLcWithoutKeyPair	
▶ 18:58:21 UTC-0500	CREATE_COMPLETE	AWS::EC2::SubnetRouteTableAssociation	PubSubnet1RouteTableAssociation	
▶ 18:58:21 UTC-0500	CREATE_COMPLETE	AWS::EC2::SubnetRouteTableAssociation	PubSubnet2RouteTableAssociation	
▶ 18:58:20 UTC-0500	CREATE_COMPLETE	AWS::EC2::SecurityGroup	EcsSecurityGroup	
▶ 18:58:19 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SecurityGroup	EcsSecurityGroup	Resource creation Initiated
▶ 18:58:06 UTC-0500	CREATE_COMPLETE	AWS::ElasticLoadBalancing::LoadBalancer	EcsElasticLoadBalancer	
▶ 18:58:06 UTC-0500	CREATE_IN_PROGRESS	AWS::ElasticLoadBalancing::LoadBalancer	EcsElasticLoadBalancer	Resource creation Initiated
▶ 18:58:06 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SubnetRouteTableAssociation	PubSubnet1RouteTableAssociation	Resource creation Initiated



Launch status

Your container instances are launching, and it may take a few minutes until they are in the running state, and ready to access. Usage hours on your new container instances will start immediately and continue to accrue until you stop or terminate.

ECS Instances status



Your instances are up and running.



Launching the requested resources using AWS CloudFormation. This may take a few AWS::EC2::InternetGateway... 15 of 15 Resources Created minutes. [Click here to view the CloudFormation stack.](#)

ECS Status



Waiting for ECS instance to launch...

This may take a few minutes



Created TaskDefinition

nginx:1



Created Service nginx-webapp successfully

The tasks will start up momentarily



Task Id	Status
No results	

[View Service](#)



Amazon ECS

Clusters

Task Definitions

Clusters > default > Service: nginx-webapp

Service : nginx-webapp

Update

Delete

Details

Cluster [default](#)

Status **ACTIVE**

Task Definition [nginx:1](#)

Desired count 1

Pending count 0

Running count 0

Service Role [ecsServiceRole](#)

Load Balancers

Load Balancer Name	Container Name	Container Port
EC2Contai-EcsElast-X6Z2SFTDEBFQ	nginx-cn	80



Tasks

Events

Deployments

Metrics

Last updated on November 10, 2015 7:00:08 PM (0m ago)



Filter in this page

Task status: **Running** Stopped

< Viewing 0-0 Tasks >

Task

Task Definition

Last status

Desired status

No results



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

Auto Scaling Groups

Create Load Balancer

Actions



Filter: EC2Contai-EcsElast-X6Z2SFTDEBFQ

1 to 1 of 1

☐	Load Balancer Name	DNS Name	Port Configuration	Availability Zones	Insta
☒	EC2Contai-EcsElast-X6Z2S...	EC2Contai-EcsElast-X6Z2S...	80 (HTTP) forwarding to 80 (...)	us-west-2b, us-west-2c	0 Insta

Load balancer: EC2Contai-EcsElast-X6Z2SFTDEBFQ

Description

Instances

Health Check

Monitoring

Security

Listeners

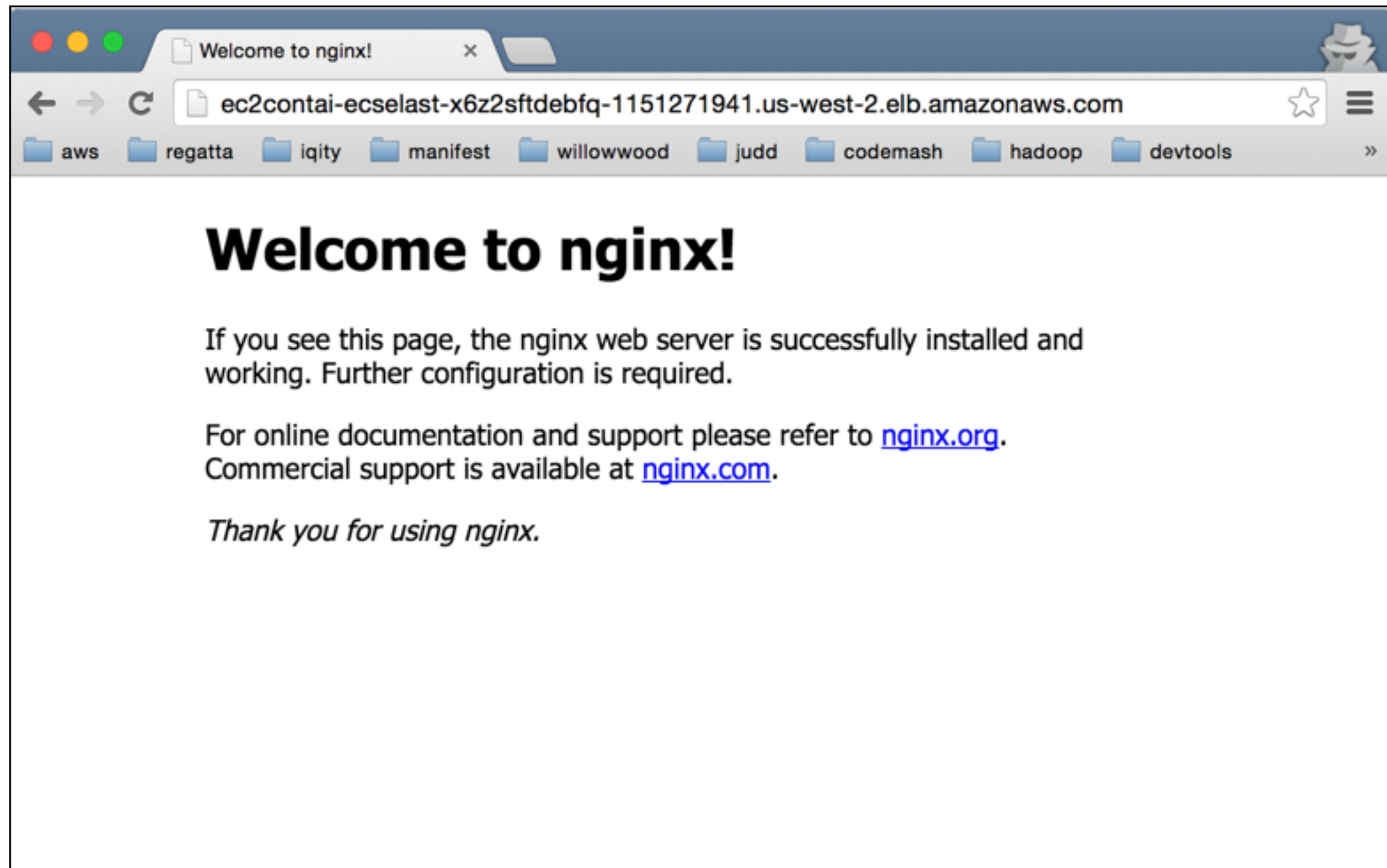
Tags

DNS Name: EC2Contai-EcsElast-X6Z2SFTDEBFQ-1151271941.us-west-2.elb.amazonaws.com (A Record)

Note: Because the set of IP addresses associated with a LoadBalancer can change over time, you should never create an "A" record with any specific IP address. If you want to use a friendly DNS name for your load balancer instead of the name generated by the Elastic Load Balancing service, you should create a CNAME record for the LoadBalancer DNS name, or use Amazon Route 53 to create a hosted zone. For more information, see [Using Domain Names With Elastic Load Balancing](#).

Scheme: internet-facing

Status: 0 of 0 instances in service



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.



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

Auto Scaling Groups

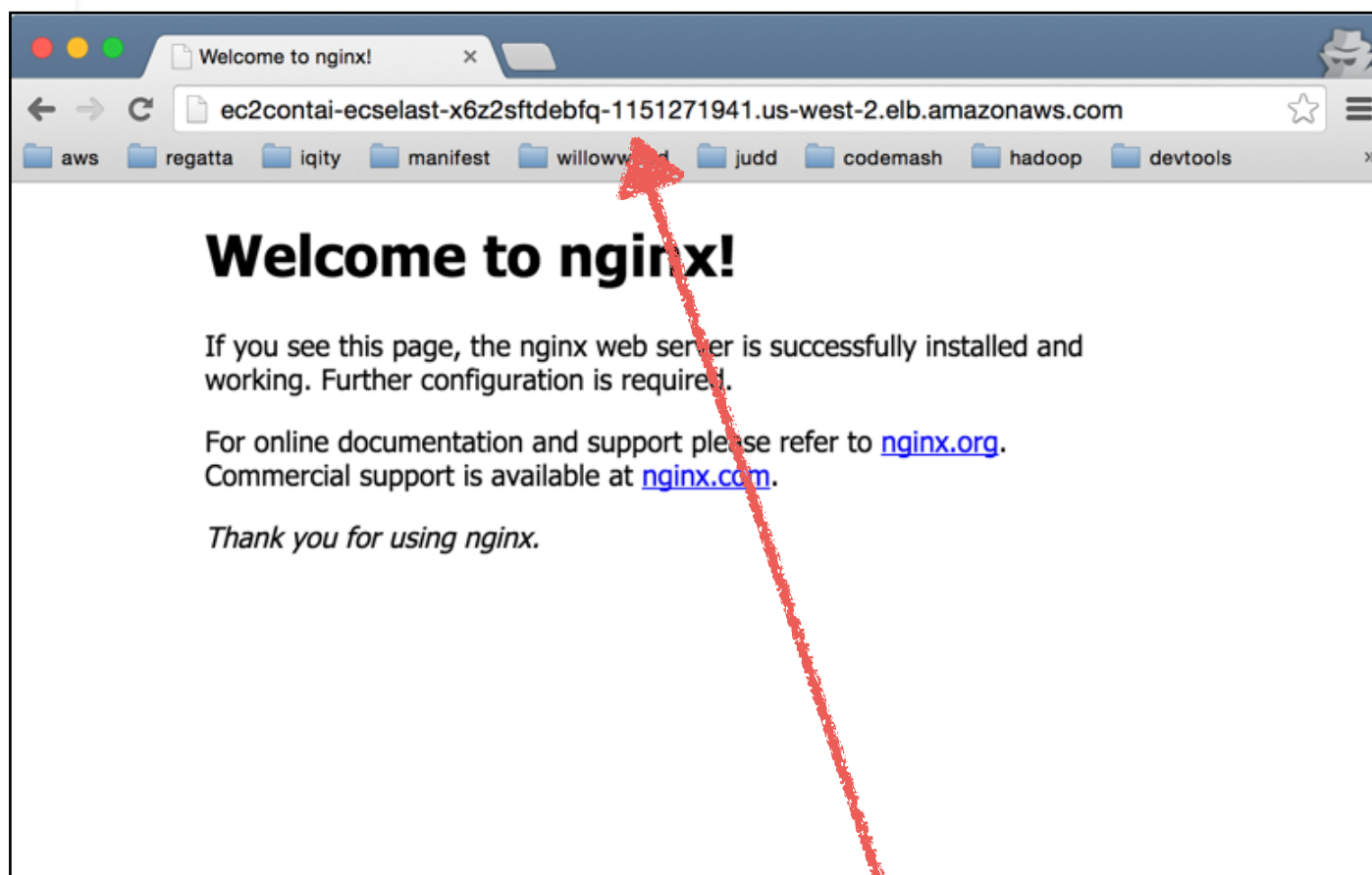
Create Load Balancer

Actions

Filter: EC2Contai-EcsElast-X6Z2SFTDEBFQ

1 to 1 of 1

☐	Load Balancer Name	DNS Name	Port Configuration	Availability Zones	Insta
☐	EC2Contai-EcsElast-X6Z2S...	EC2Contai-EcsElast-X6Z2S...	80 (HTTP) forwarding to 80 (...)	us-west-2b, us-west-2c	0 Insta



Load balancer: EC2Contai-EcsElast-X6Z2SFTDEBFQ

Description

Instances

Health Check

Monitoring

Security

Listeners

Tags

DNS Name: EC2Contai-EcsElast-X6Z2SFTDEBFQ-1151271941.us-west-2.elb.amazonaws.com (A Record)

Note: Because the set of IP addresses associated with a LoadBalancer can change over time, you should never create an "A" record with any specific IP address. If you want to use a friendly DNS name for your load balancer instead of the name generated by the Elastic Load Balancing service, you should create a CNAME record for the LoadBalancer DNS name, or use Amazon Route 53 to create a hosted zone. For more information, see [Using Domain Names With Elastic Load Balancing](#).

Scheme: internet-facing

Status: 0 of 0 instances in service



Amazon ECS

Clusters

Task Definitions

Clusters > default > Task: 14228d47-cb18-4d90-aedb-870ed40ac548

Task : 14228d47-cb18-4d90-aedb-870ed40ac548

Run more like this

Stop

Details

Cluster	default
Container Instance	db69e17e-a9eb-4d4f-9e49-52b23b1d632e
EC2 Instance Id	i-73a94fa9
Task Definition	nginx:1
Last Status	RUNNING
Desired Status	RUNNING

Containers

Last updated on November 10, 2015 7:24:18 PM (0m ago)



Name	Container Id	Status	Image	CPU...	Me...	Ess...
nginx-cn	5a15c52c-8e4e-4089-8651-788de296ee4b	RUN...	nginx	0	512	true

Details

Exit Code 0

Network bindings

Host Port	Container Port	Protocol	External Link
80	80	tcp	52.11.106.96:80

Environment variables

Key	Value
No Environment Variables	

Docker labels

Key	Value
No docker labels	

Extra hosts

Hostname	IP address
No host entries	

Mount points

Container Path	Source Volume	Read only
No Mount Points		

Volumes from

Source Container	Read only
No volumes from	

Ulimits

Name	Soft limit	Hard limit
No ulimit		

Log Configuration

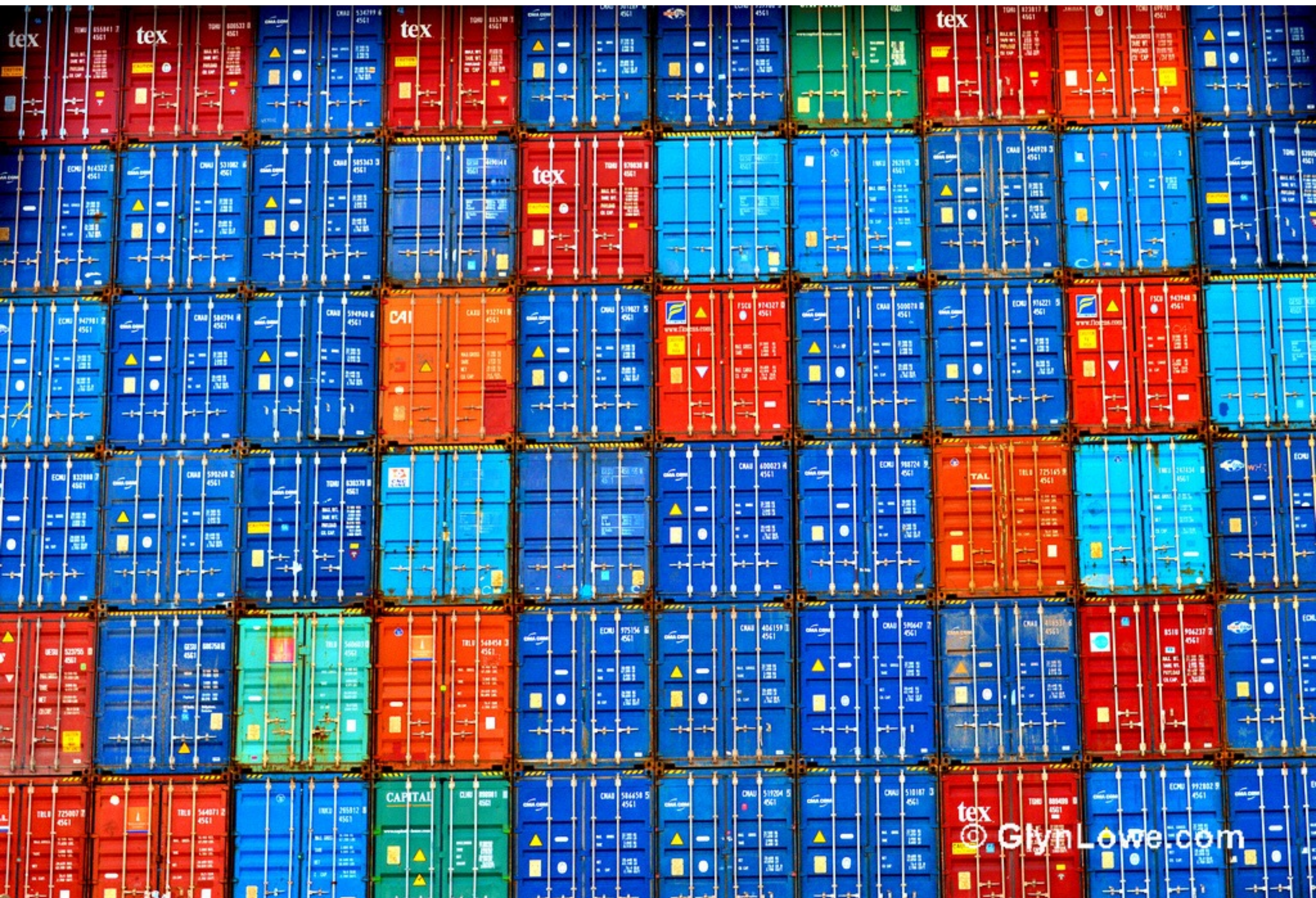
Log driver:

Key	Value
No log configuration	

SUMMARY



<https://github.com/coreos/rkt>

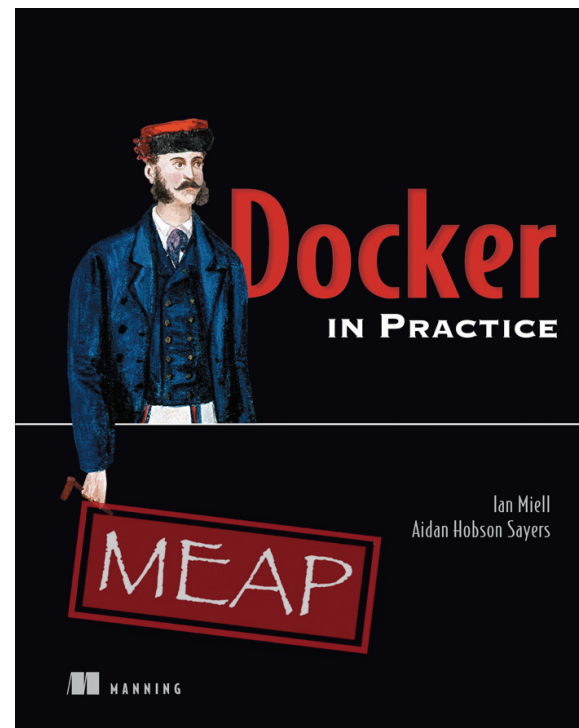
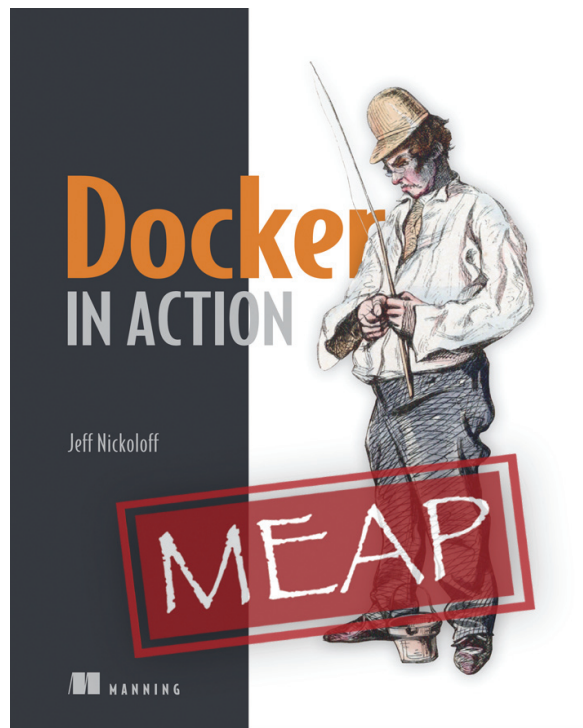
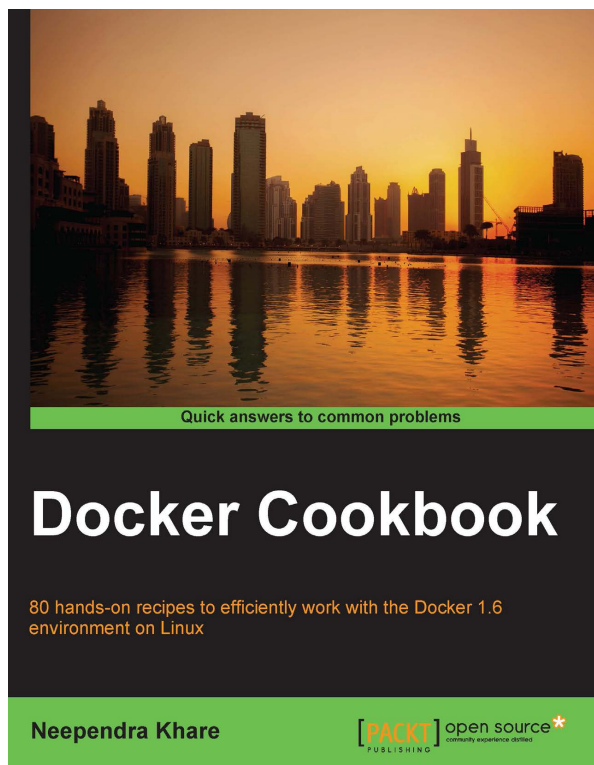




WELCOME
TO THE BARREL
Please eat
yourself a beer
A sign with a logo

MEET THE BREWERY:
Odell Brewing Co
(For culture)
Pilsner, IPA, etc.
Brewery Co

RESOURCES



221

DZone
RECARDZ

BROUGHT TO YOU BY

Site24x7

CONTENTS

- About Docker
- Docker Architecture
- Getting Started
- Typical Local Workflow
- Other Useful Commands
- Dockerfile, and more...

Getting Started With Docker

By Christopher M. Judd

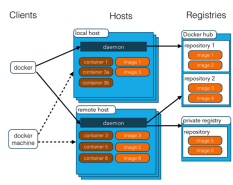
ABOUT DOCKER

Almost overnight, Docker has become the de facto standard that developers and system administrators use for packaging, deploying, and running distributed 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.

Similar to the popular version control software Git, Docker has a social aspect, in that developers and sysadmins are able to share their images via [Docker Hub](#).

Docker is an open-source solution that runs natively on Linux but also works on Windows and Mac using a lightweight Linux distribution and VirtualBox. Many tools have also grown up around Docker to make it easier to manage and orchestrate complex distributed applications.



The diagram illustrates the Docker architecture. On the left, 'Clients' (Docker and Docker Machine) connect to 'Hosts' (Docker Hosts). The hosts contain 'Containers' (e.g., container 1, container 2, container 3) and 'Images' (e.g., image 1, image 2, image 3). On the right, 'Registries' (Docker Hub, Registry 1, Registry 2, Private registry) are shown. Arrows indicate the flow of images and containers between these components.

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 originated.

Docker Client
Command-line utility or other tool that takes advantage of the Docker API (https://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.

Site24x7
DOCKER MONITORING
Get Detailed Insight into Docker Containers



LEARN MORE >>

© DZONE, INC.

DZONE.COM



Thanks to Brian Sherwin
for making the presentation
more Windows friendly

@bsherwin