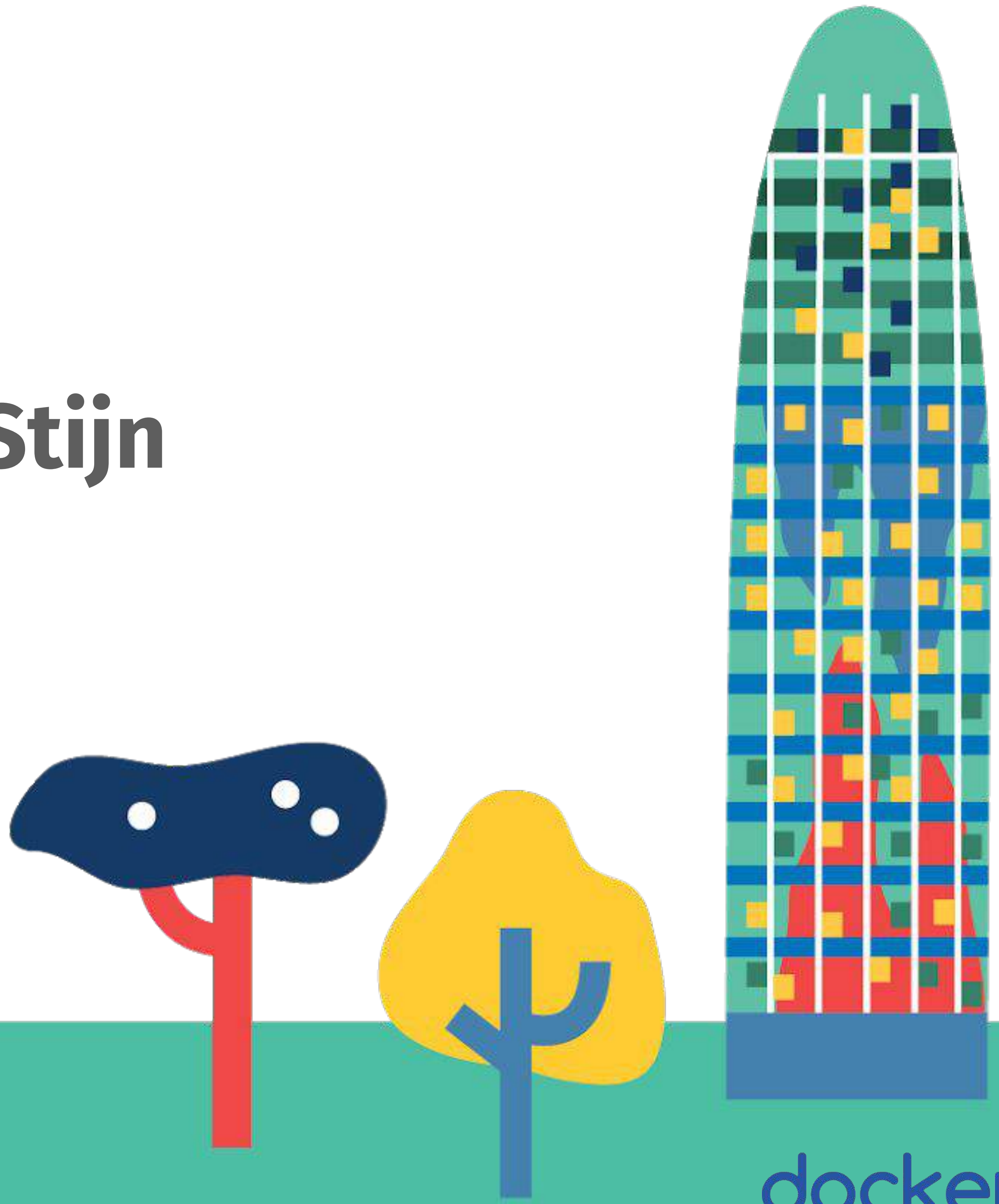
Dockerfile Best Practices

Tibor Vass
Docker, Inc.
@tiborvass

Sebastiaan van Stijn
Docker, Inc.
@thaJeztah

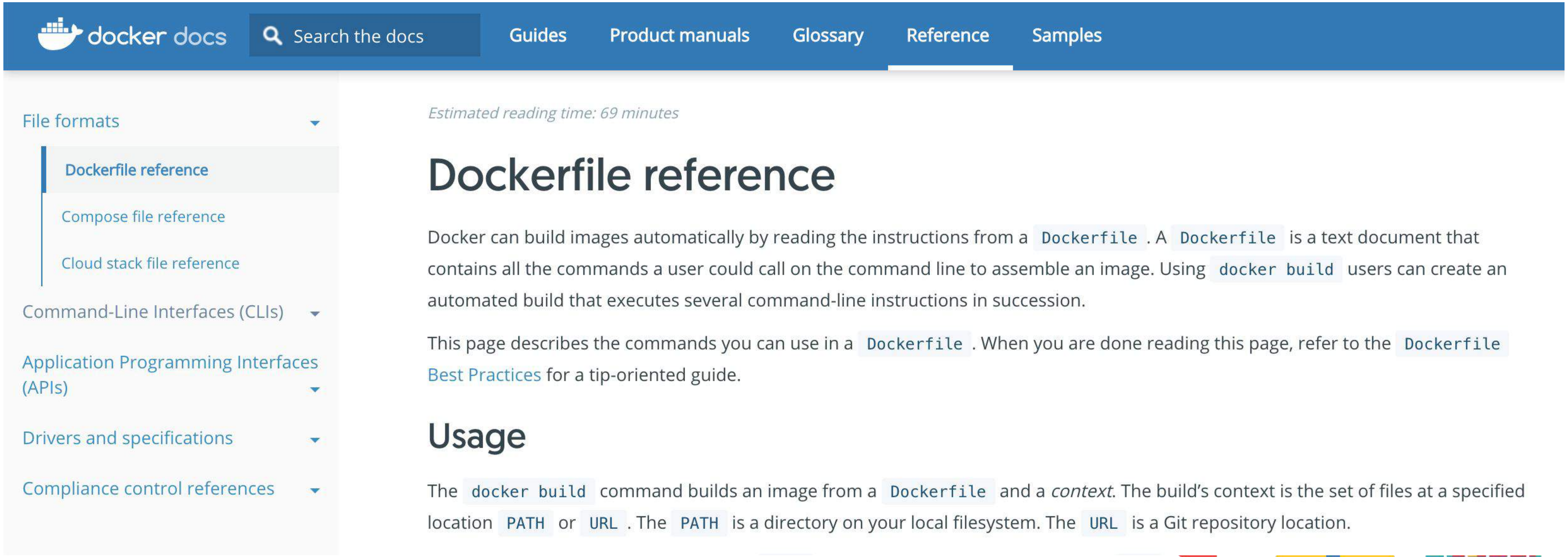


Dockerfile

A blueprint to build Docker images

Popular: 1+ million Dockerfiles on GitHub

https://docs.docker.com/engine/reference/builder/


docker docs
Search the docs
Guides
Product manuals
Glossary
Reference
Samples
File formats
Dockerfile reference
Compose file reference
Cloud stack file reference
Command-Line Interfaces (CLIs)
Application Programming Interfaces (APIs)
Drivers and specifications
Compliance control references

Estimated reading time: 69 minutes

Dockerfile reference

Docker can build images automatically by reading the instructions from a `Dockerfile`. A `Dockerfile` is a text document that contains all the commands a user could call on the command line to assemble an image. Using `docker build` users can create an automated build that executes several command-line instructions in succession.

This page describes the commands you can use in a `Dockerfile`. When you are done reading this page, refer to the `Dockerfile` Best Practices for a tip-oriented guide.

Usage

The `docker build` command builds an image from a `Dockerfile` and a *context*. The build's context is the set of files at a specified location `PATH` or `URL`. The `PATH` is a directory on your local filesystem. The `URL` is a Git repository location.

Use latest Docker, enable BuildKit today!

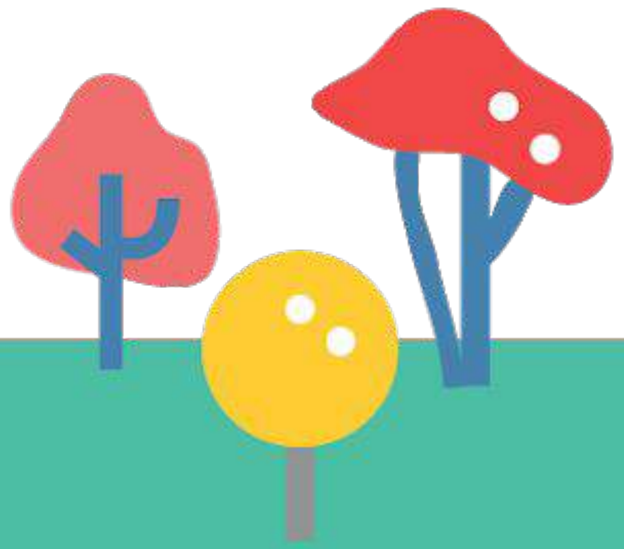
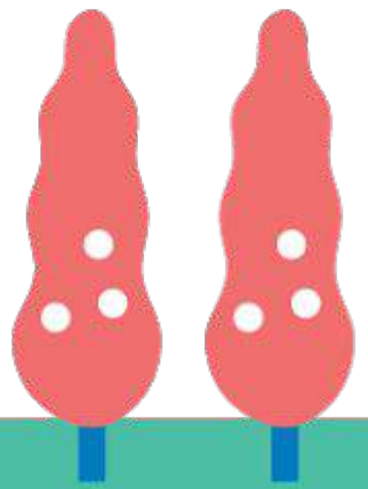
Docker client:

```
export DOCKER_BUILDKIT=1
```

Windows support coming soon

Docker daemon config:

```
{
  "features": {"buildkit": true}
}
```

Quick refresher

Quick refresher on Images

image: template to instantiate running containers.
References list of filesystem layers

layer: a list of changes to a rootfs

copy-on-write filesystem: allows smaller disk usage

Quick refresher on Build

Parse Dockerfile and get build steps to perform

build caching: no need to perform build steps where files or RUN line have not changed, reuse cached layers

build context: local files that can be copied to the image

Improving Dockerfiles

Areas of improvements

- Consistency/Repeatability
- (Incremental) build time
- Image size
- Maintainability

Example project

Basic Java Spring Hello world web app

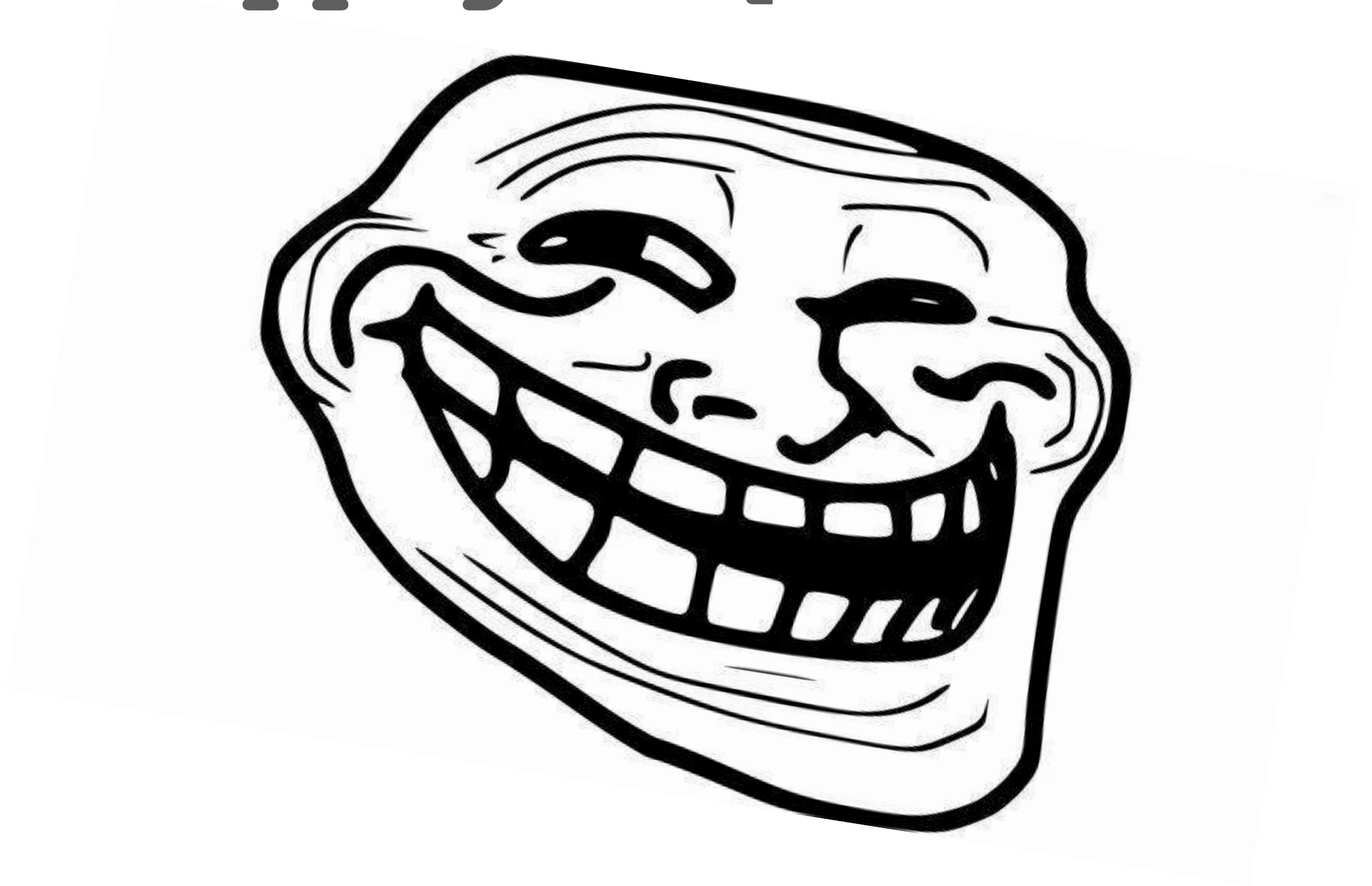
```
-rw-r--r-- 1   656 Dec  4 12:20 Dockerfile
drwxr-xr-x 2  6.1M Dec  4 09:44 docs/
-rw-r--r-- 1  1.7K Dec  3 09:48 pom.xml
-rw-r--r-- 1  1.0K Dec  4 10:12 README.md
drwxr-xr-x 4   44K Dec  3 09:48 src/
drwxr-xr-x 2   17M Dec  4 09:50 target/
```

Let’s improve this Dockerfile

```
FROM debian
COPY . /app
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh emacs
CMD ["java", "-jar", "/app/target/app.jar"]
```

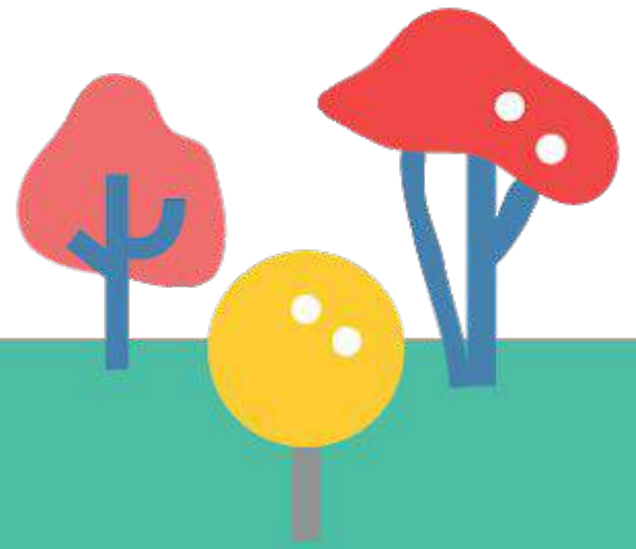
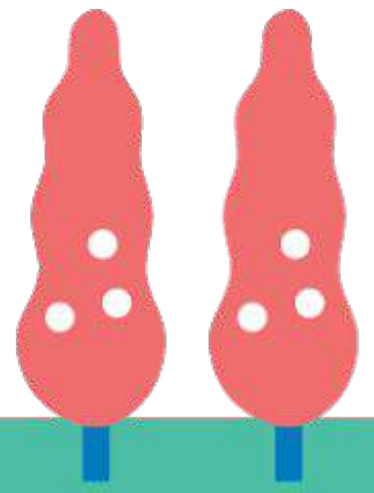
Let’s improve this Dockerfile

```
FROM debian
COPY . /app
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh ~~emacs~~ vim
CMD ["java", "-jar", "/app/target/app.jar"]
```



Order matters for caching

```
FROM debian
~~COPY . /app~~
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY . /app
CMD ["java", "-jar", "/app/target/app.jar"]
```

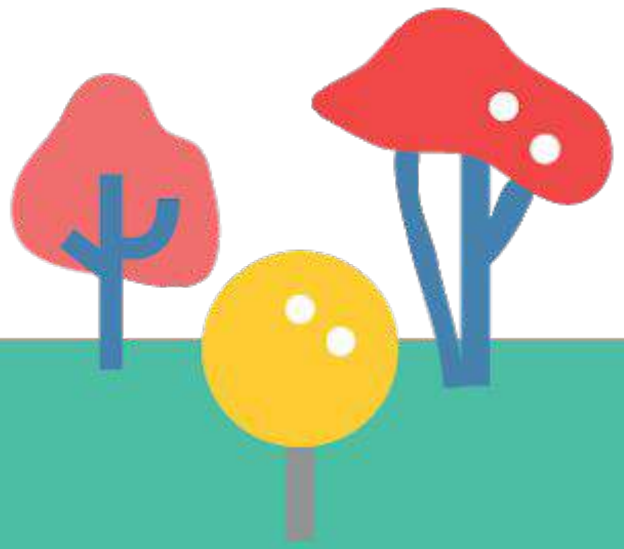
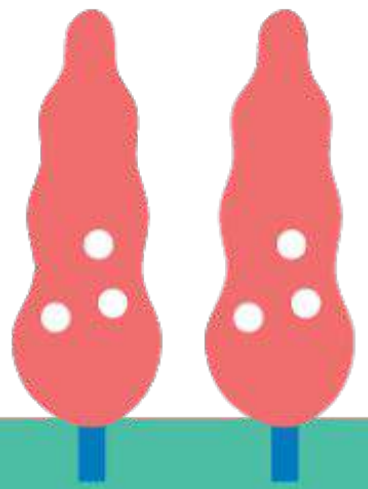



Order matters for caching

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY . /app
CMD ["java", "-jar", "/app/target/app.jar"]
```

More specific COPY to limit cache bust

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
~~COPY . /app~~
COPY target/app.jar /app
CMD ["java", "-jar", "/app~~/target~~/app.jar"]
```

More specific COPY to limit cache bust

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Pro Tip! Use `COPY`, not `ADD` for local files

More specific COPY to limit cache bust

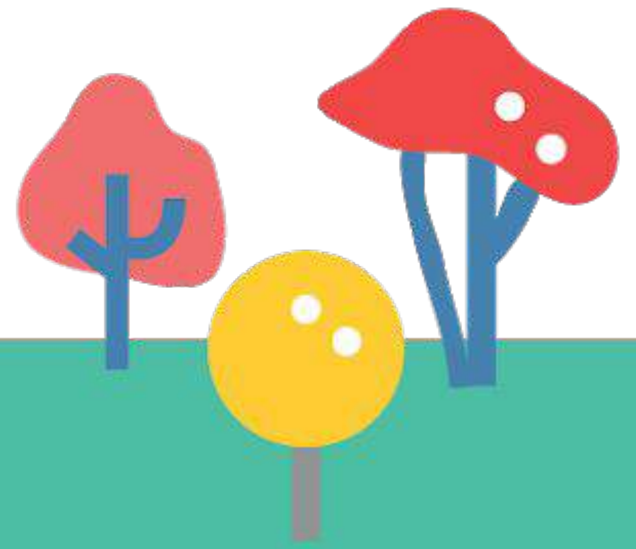
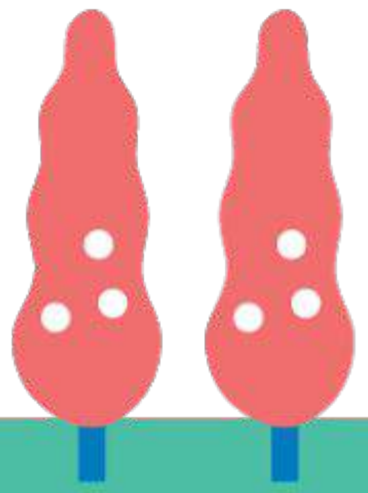
```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Identify cacheable "units"

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Line buddies: apt-get update & install

```
FROM debian
~~RUN apt-get update~~
~~RUN apt-get -y install openjdk-8-jdk ssh vim~~
RUN apt-get update && apt-get -y install \
    openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Line buddies: apt-get update & install

```
FROM debian
RUN apt-get update && apt-get -y install \
    openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Remove unnecessary dependencies

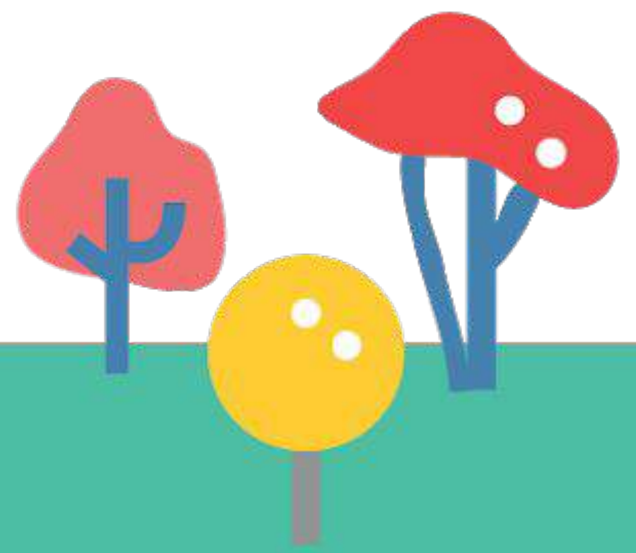
```
FROM debian
RUN apt-get update && apt-get -y install \
    openjdk-8-jdk~~ ssh vim~~
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

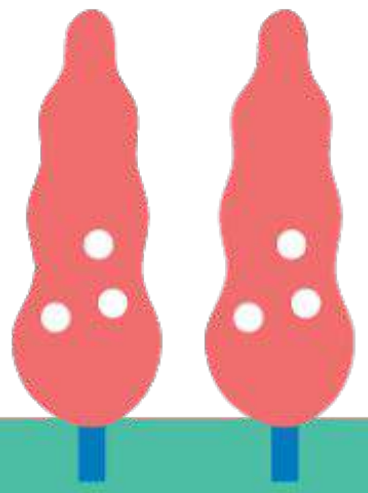
Remove unnecessary dependencies

```
FROM debian
RUN apt-get update && apt-get -y install \
    openjdk-8-jdk
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Use --no-install-recommends

```
FROM debian
RUN apt-get update && \
    apt-get -y install --no-install-recommends \
    openjdk-8-jdk
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```





Remove package manager cache

```
FROM debian
RUN apt-get update && \
    apt-get -y install --no-install-recommends \
    openjdk-8-jdk \
    && rm -rf /var/lib/apt/lists/*
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Remove package manager cache

```
FROM debian
RUN apt-get update && \
    apt-get -y install --no-install-recommends \
    openjdk-8-jdk \
    && rm -rf /var/lib/apt/lists/*
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Reuse official images when possible

```
~~FROM debian~~
~~RUN apt-get update && \~~
    ~~apt-get -y install --no-install-recommends \~~
    ~~openjdk-8-jdk\~~
    ~~&& rm -rf /var/lib/apt/lists/*~~
FROM openjdk
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Reuse official images when possible

- Reduce time spent on maintenance (frequently updated with fixes)
- Reduce size (shared layers between images)
- Pre-configured for container use
- Built by smart people 🤗
- Bonus: scanned for vulnerabilities on Docker Hub

Reuse official images when possible

```
FROM openjdk
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

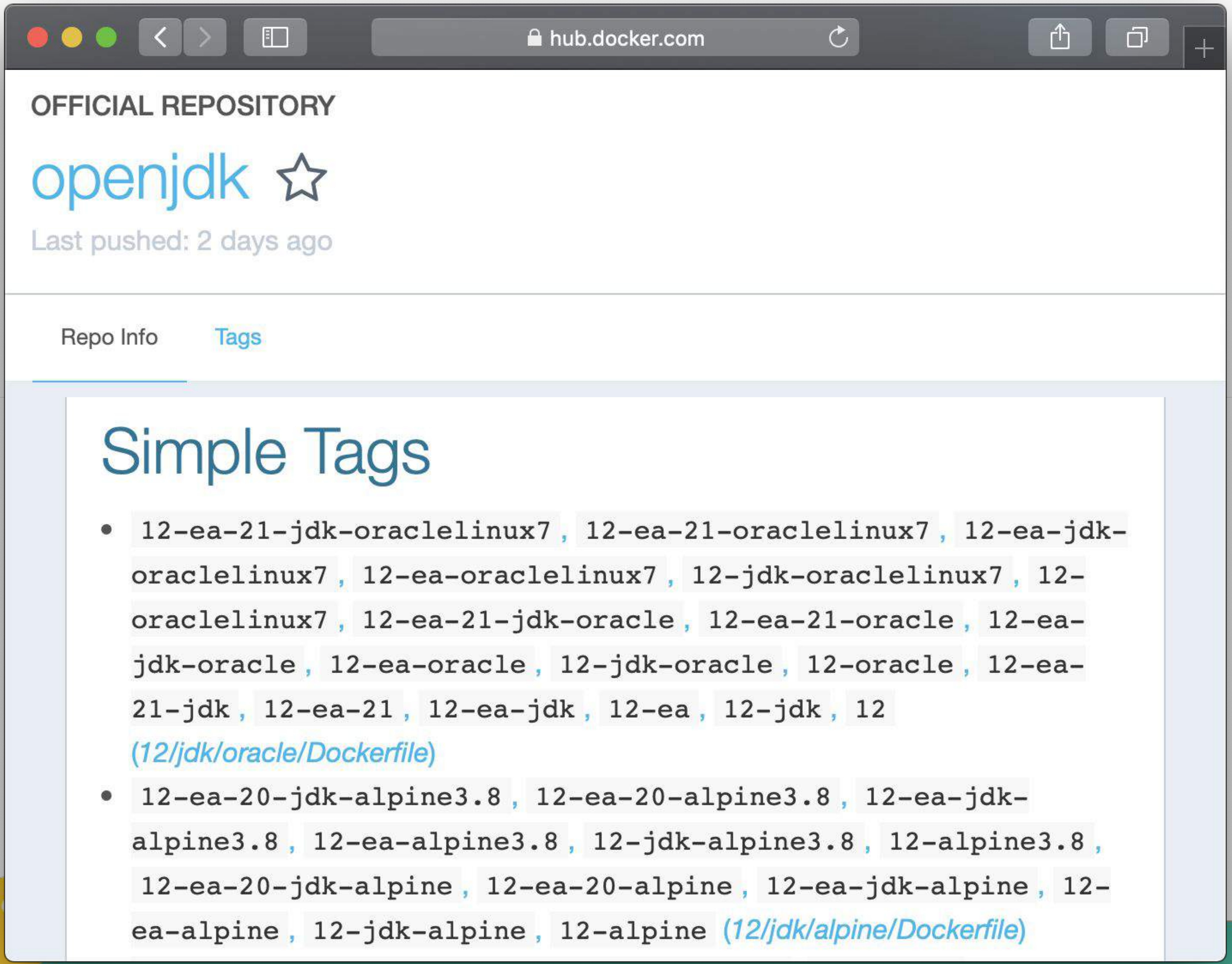
Use more specific tags

```
~~FROM openjdk:latest~~
FROM openjdk:8
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Use more specific tags

```
FROM openjdk:8
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Use more specific tags



hub.docker.com

OFFICIAL REPOSITORY

openjdk

Last pushed: 2 days ago

Repo Info | Tags

Simple Tags

- 12-ea-21-jdk-oraclelinux7 , 12-ea-21-oraclelinux7 , 12-ea-jdk-oraclelinux7 , 12-ea-oraclelinux7 , 12-jdk-oraclelinux7 , 12-oraclelinux7 , 12-ea-21-jdk-oracle , 12-ea-21-oracle , 12-ea-jdk-oracle , 12-ea-oracle , 12-jdk-oracle , 12-oracle , 12-ea-21-jdk , 12-ea-21 , 12-ea-jdk , 12-ea , 12-jdk , 12 (12/jdk/oracle/Dockerfile)
- 12-ea-20-jdk-alpine3.8 , 12-ea-20-alpine3.8 , 12-ea-jdk-alpine3.8 , 12-ea-alpine3.8 , 12-jdk-alpine3.8 , 12-alpine3.8 , 12-ea-20-jdk-alpine , 12-ea-20-alpine , 12-ea-jdk-alpine , 12-ea-alpine , 12-jdk-alpine , 12-alpine (12/jdk/alpine/Dockerfile)

Read the image's documentation on Docker Hub

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

Use more specific tags

```
FROM openjdk:8-jre
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Use more specific tags

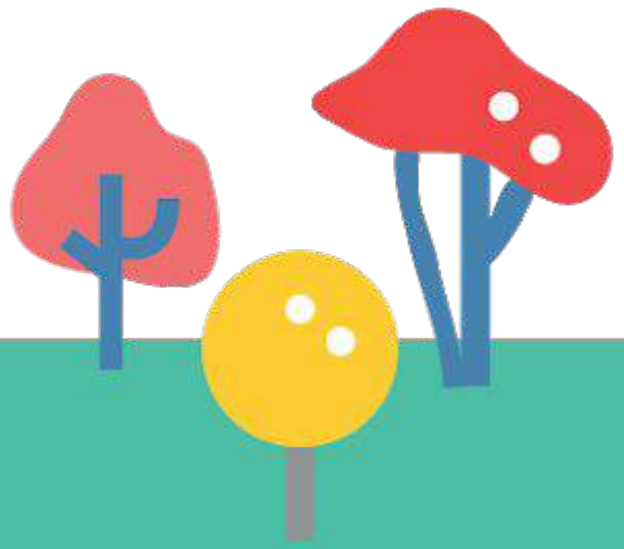
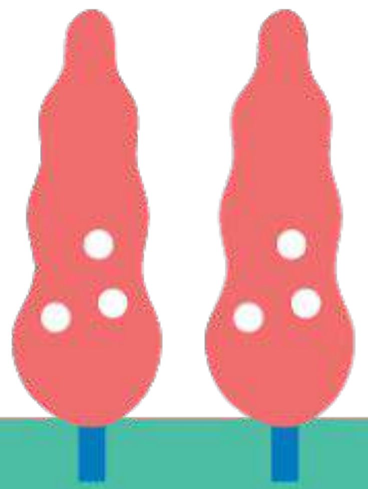
```
FROM openjdk:8-jre
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Look for minimal flavors

```
FROM openjdk:8-jre-slim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Look for minimal flavors

```
FROM openjdk:8-jre~~-slim~~
FROM openjdk:8-jre-alpine
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Look for minimal flavors

```
REPOSITORY    TAG               SIZE
openjdk       8                 624MB
openjdk       8-jre             443MB
openjdk       8-jre-slim        204MB
openjdk       8-jre-alpine      83MB
```

Look for minimal flavors

```
FROM openjdk:8-jre-alpine
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Look for reproducibility

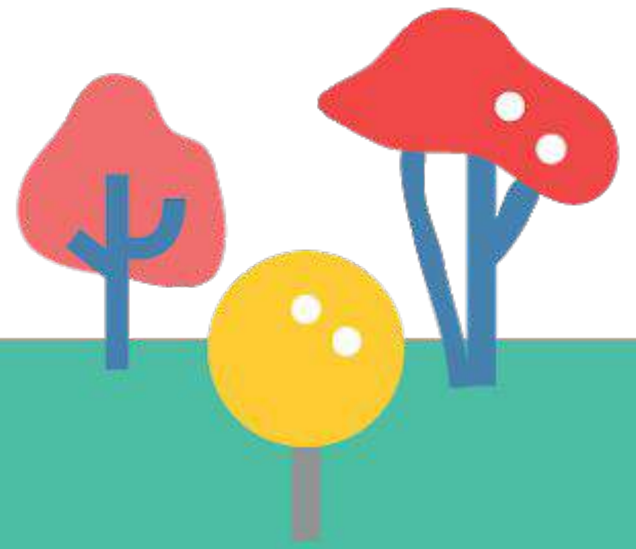
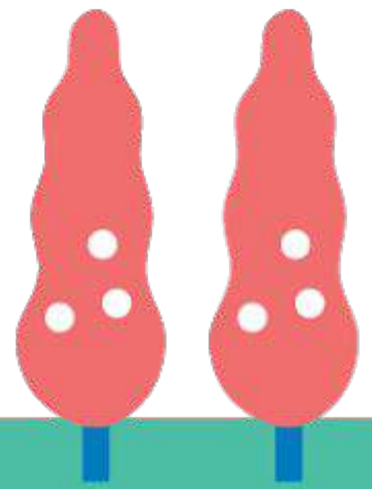
```
FROM openjdk:8-jre-alpine
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Build from source in a consistent environment

- Build environment is described in the Dockerfile
- Correct versions of build tools installed
- Prevent inconsistencies between environments
- There may be system dependencies
- *The "source of truth" is the source code not the build artifact*

Build from source in a consistent environment

```
~~FROM openjdk:8-jre-alpine~~
FROM maven:3.6-jdk-8-alpine
~~COPY app.jar /app~~
COPY pom.xml /app/
COPY src /app/src
RUN cd /app && mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

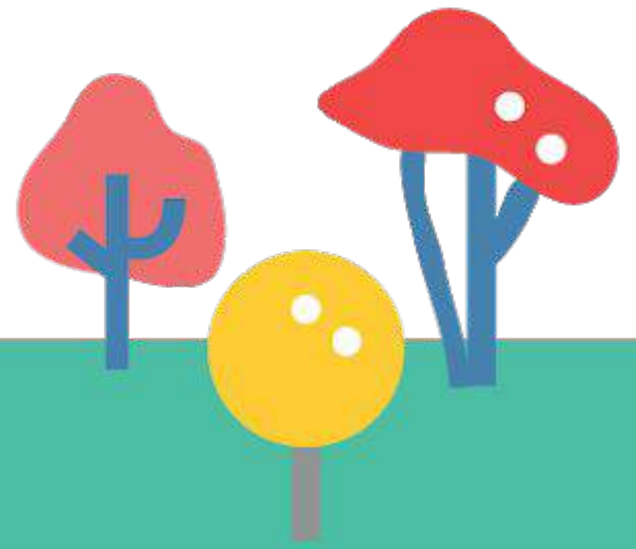
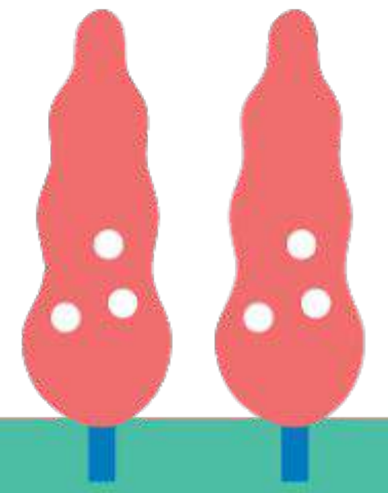



Build from source in a consistent environment

```
FROM maven:3.6-jdk-8-alpine
COPY pom.xml /app/
COPY src /app/src
RUN cd /app && mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Build from source in a consistent environment

```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml ~~/app/~~.
COPY src ~~/app~~./src
RUN ~~cd /app &&~~ mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Build from source in a consistent environment

```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Cache dependencies

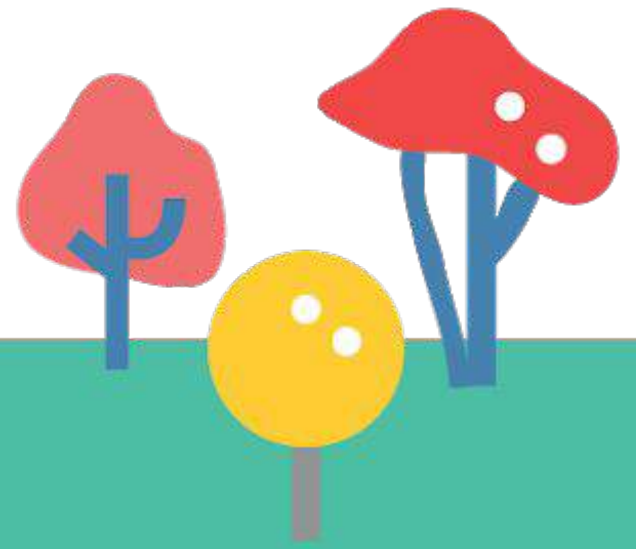
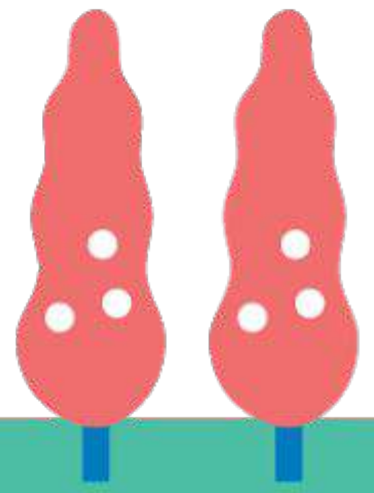
```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Cache dependencies

```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Identify build dependencies

```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Multi-stage builds to remove build deps

```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
```

Multi-stage builds to remove build deps

```
FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
~~CMD ["java", "-jar", "/app/app.jar"]~~

FROM openjdk:8-jre-alpine
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

Multi-stage builds to remove build deps

```
FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package

FROM openjdk:8-jre-alpine
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

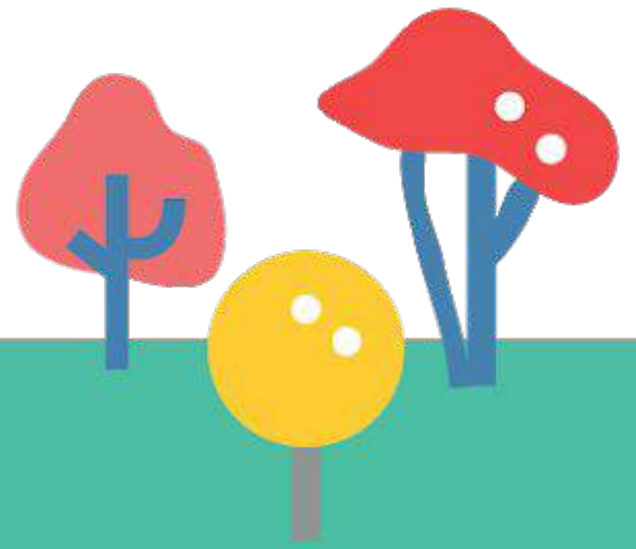
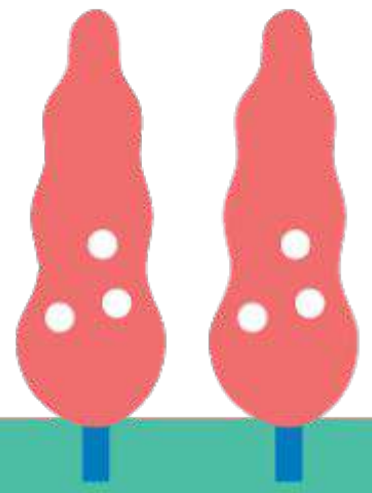
Projects with many stages

- Moby: 16 stages
  https://github.com/moby/moby/blob/master/Dockerfile

- BuildKit: 44 stages
  https://github.com/moby/buildkit/blob/master/hack/dockerfiles/test.buildkit.Dockerfile

Multi-stage usecases

- Separate build from runtime environment (shrinking image size)
- Slight variations on images
- DRY (Don't Repeat Yourself)
- Build/dev/test/lint environments
- Concurrent stages
- Platform-specific stages

docker build --target X

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-jessie AS release-jessie
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]

FROM openjdk:8-jre-alpine AS release-alpine
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

docker build --target X

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-jessie AS release-jessie
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]

FROM openjdk:8-jre-alpine AS release-alpine
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

docker build --target X

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-jessie AS release-jessie
COPY --from=builder /app/app.jar /
CMD ["java", "-jar", "/app.jar"]

FROM openjdk:8-jre-alpine AS release-alpine
COPY --from=builder /app/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

Global ARG: docker build --build-arg K=V

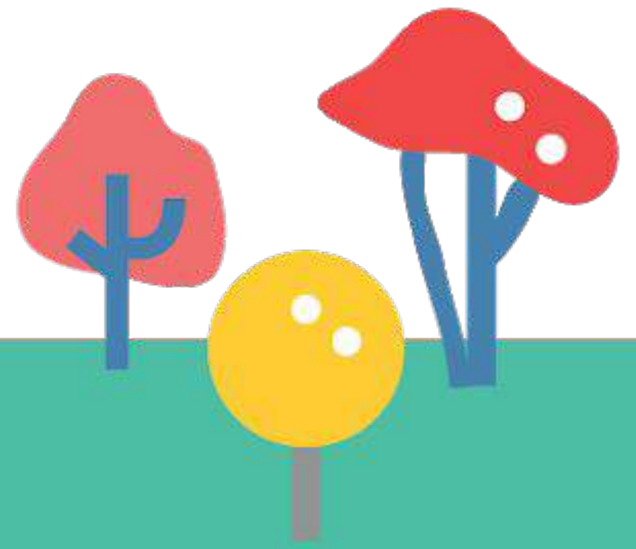
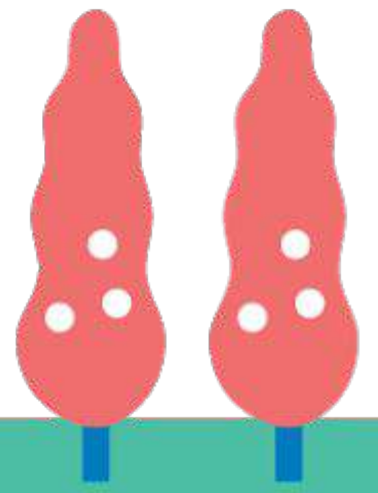
```
ARG flavor=alpine

FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-$flavor AS release
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

Various environments: build, dev, test, lint, ...

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-alpine AS lint
RUN wget https://github.com/checkstyle/checkstyle/releases/download/checkstyle-8.15/checkstyle-8.15-all.jar
COPY checks.xml .
COPY src /src
RUN java -jar checkstyle-8.15-all.jar -c checks.xml /src
```

Various environments: build, dev, test, lint, ...

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM openjdk:8-jre-alpine AS release
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]

FROM builder AS dev
RUN apk add --no-cache strace
ENTRYPOINT ["ash"]
```

Various environments: build, dev, test, lint, …

```
FROM maven:3.6-jdk-8-alpine AS builder
...
RUN mvn -e -B package -DskipTests

FROM builder AS unit-test
RUN mvn -e -B test

FROM release AS integration-test
RUN apk add --no-cache curl
RUN ./test/run.sh
```

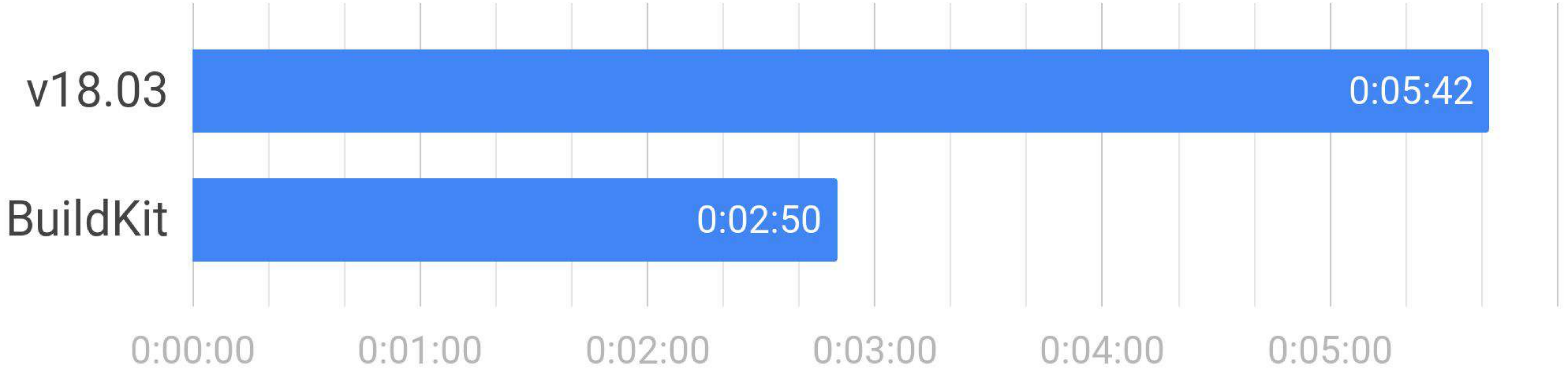
Multi-stage: build concurrently

```
FROM maven:3.6-jdk-8-alpine AS builder
...

FROM tiborvass/whalesay AS assets
RUN whalesay "¡Hola DockerCon!" > /out/assets.html

FROM openjdk:8-jre-alpine AS release
COPY --from=builder /app/app.jar /
COPY --from=assets /out /assets
CMD ["java", "-jar", "/app.jar"]
```

Benchmarks

Based on **github.com/moby/moby** Dockerfile, master branch. **Smaller** is better.

Time for full build from empty state

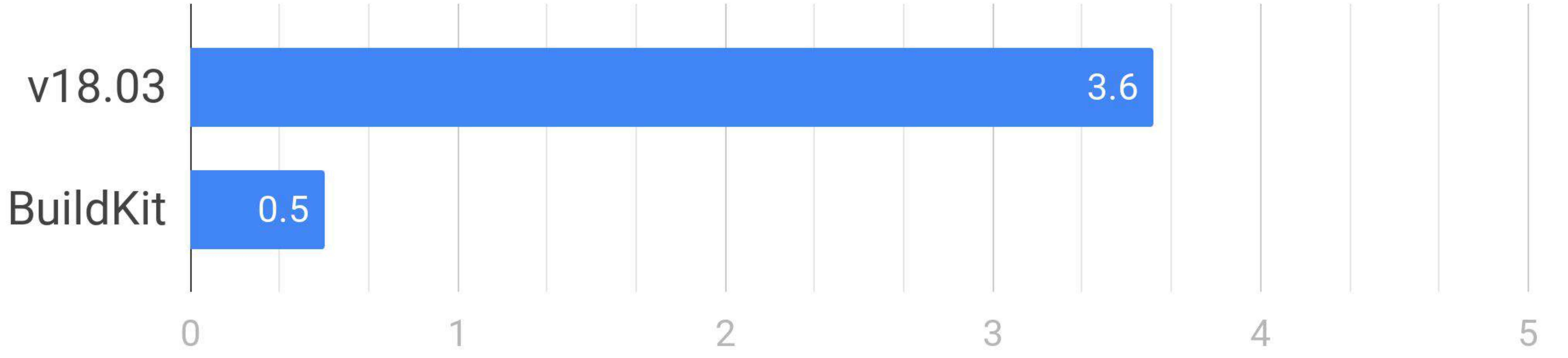


2.0x faster

Benchmarks

Based on **github.com/moby/moby** Dockerfile, master branch. **Smaller** is better.

Repeated build with matching cache

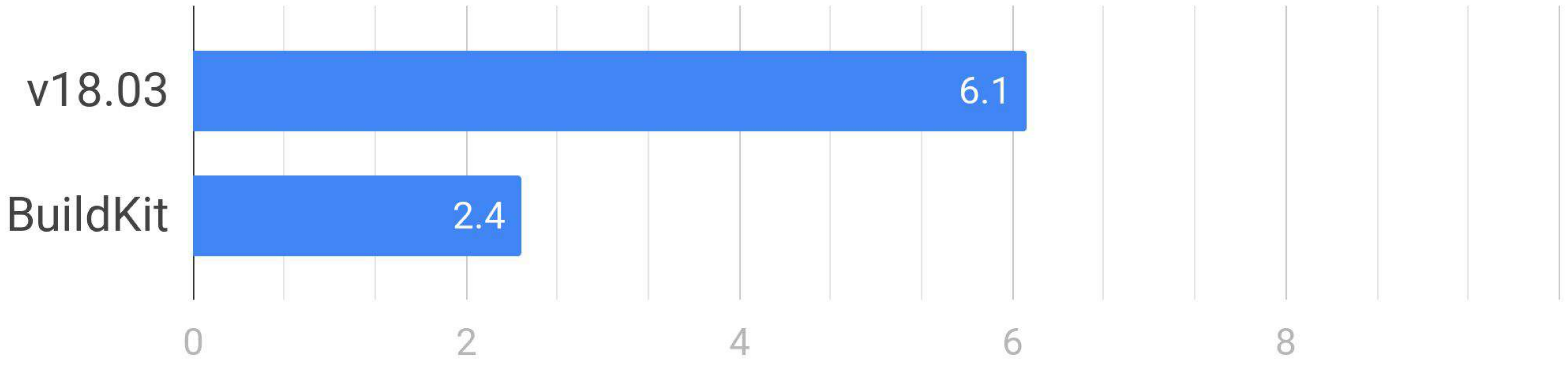


7.2x faster

Benchmarks

Based on **github.com/moby/moby** Dockerfile, master branch. **Smaller** is better.

Repeated build with new source code



2.5x faster

Some new Dockerfile features in v18.09

“Supercharged Docker Build with BuildKit”

BlackBelt session ***Wednesday 12pm***

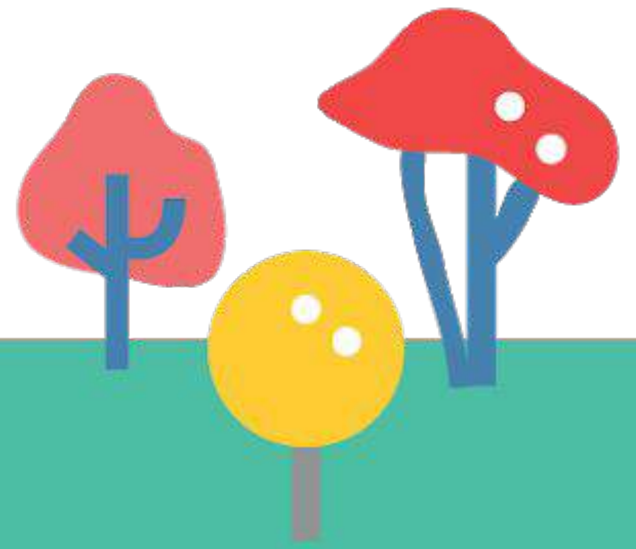
- What’s new
- New Dockerfile features (RUN --mount, secrets, ssh, syntax customization)

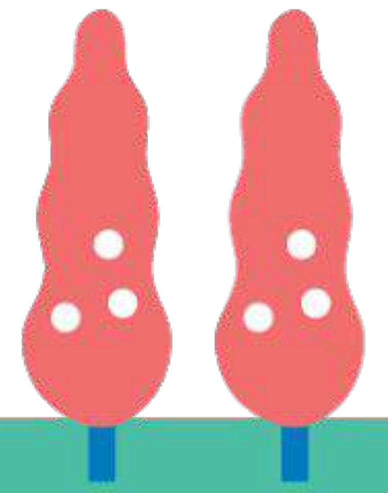
syntax = docker/dockerfile:1.0-experimental

```
# syntax=docker/dockerfile:1.0-experimental

FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
COPY . /app
RUN mvn -e -B package

FROM openjdk:8-jre-alpine
COPY --from=builder /app/app.jar /
CMD ["java", "-jar", "/app.jar"]
```

Context mounts (v18.09 only)

```
# syntax=docker/dockerfile:1.0-experimental

FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
~~COPY . /app~~
RUN --mount=target=. mvn -e -B package -DoutputDirectory=/

FROM openjdk:8-jre-alpine
COPY --from=builder /app~~/app~~.jar /
CMD ["java", "-jar", "/app.jar"]
```





Context mounts (v18.09 only)

```
# syntax=docker/dockerfile:1.0-experimental

FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
RUN --mount=target=. mvn -e -B package -DoutputDirectory=/

FROM openjdk:8-jre-alpine
COPY --from=builder /app.jar /
CMD ["java", "-jar", "/app.jar"]
```

Application cache (v18.09 only)

```
# syntax=docker/dockerfile:1.0-experimental

FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
RUN --mount=target=. --mount=type=cache,target=/root/.m2 \
    && mvn package -DoutputDirectory=/

FROM openjdk:8-jre-alpine
COPY --from=builder /app.jar /
CMD ["java", "/app.jar"]
```

Improvements recap

We went from:

- inconsistent build/dev/test environments
- bloated image
- slow build and incremental build times (cache busts)

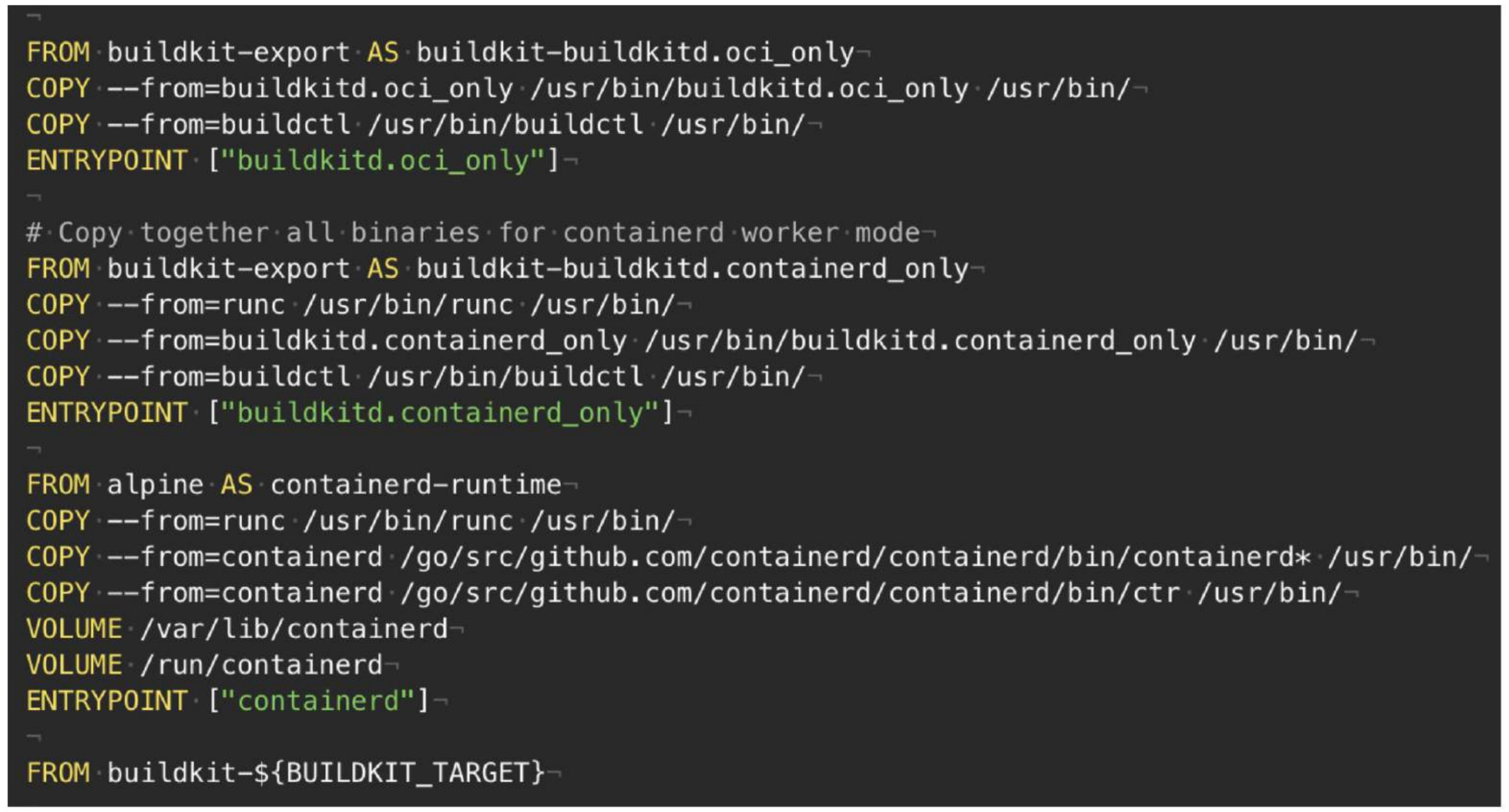
To:

- consistent build/dev/test environments
- minimal image
- very fast build and incremental build times

Read more on blog posts

Tõnis Tiigi Follow
Jun 4 · 6 min read

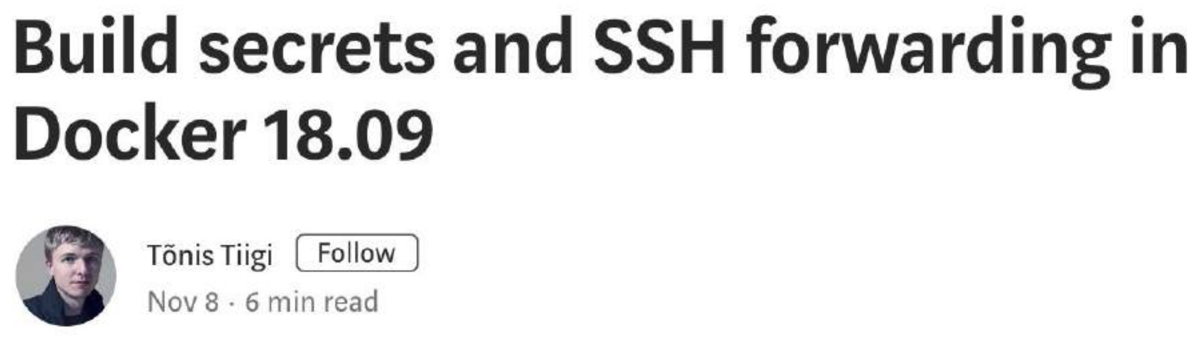
Advanced multi-stage build patterns



```
FROM buildkit-export AS buildkit-buildkitd.oci_only
COPY --from=buildkitd.oci_only /usr/bin/buildkitd.oci_only /usr/bin/
COPY --from=buildctl /usr/bin/buildctl /usr/bin/
ENTRYPOINT ["buildkitd.oci_only"]

# Copy together all binaries for containerd worker mode
FROM buildkit-export AS buildkit-buildkitd.containerd_only
COPY --from=runc /usr/bin/runc /usr/bin/
COPY --from=buildkitd.containerd_only /usr/bin/buildkitd.containerd_only /usr/bin/
COPY --from=buildctl /usr/bin/buildctl /usr/bin/
ENTRYPOINT ["buildkitd.containerd_only"]

FROM alpine AS containerd-runtime
COPY --from=runc /usr/bin/runc /usr/bin/
COPY --from=containerd /go/src/github.com/containerd/containerd/bin/containerd* /usr/bin/
COPY --from=containerd /go/src/github.com/containerd/containerd/bin/ctr /usr/bin/
VOLUME /var/lib/containerd
VOLUME /run/containerd
ENTRYPOINT ["containerd"]

FROM buildkit-${BUILDKIT_TARGET}
```

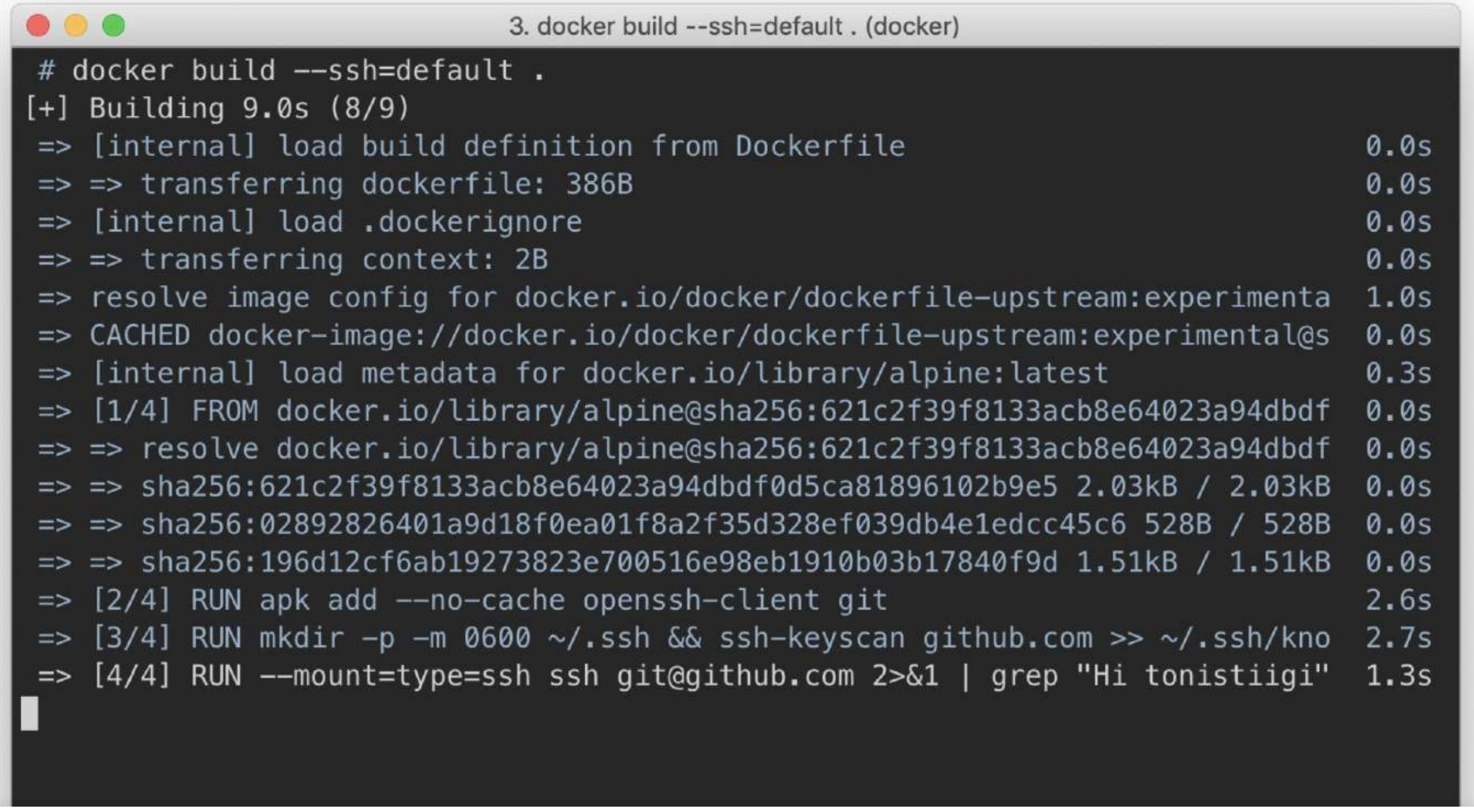
https://medium.com/@tonistiigi/advanced-multi-stage-build-patterns-6f741b852fae

https://medium.com/@tonistiigi/build-secrets-and-ssh-forwarding-in-docker-18-09-ae8161d066



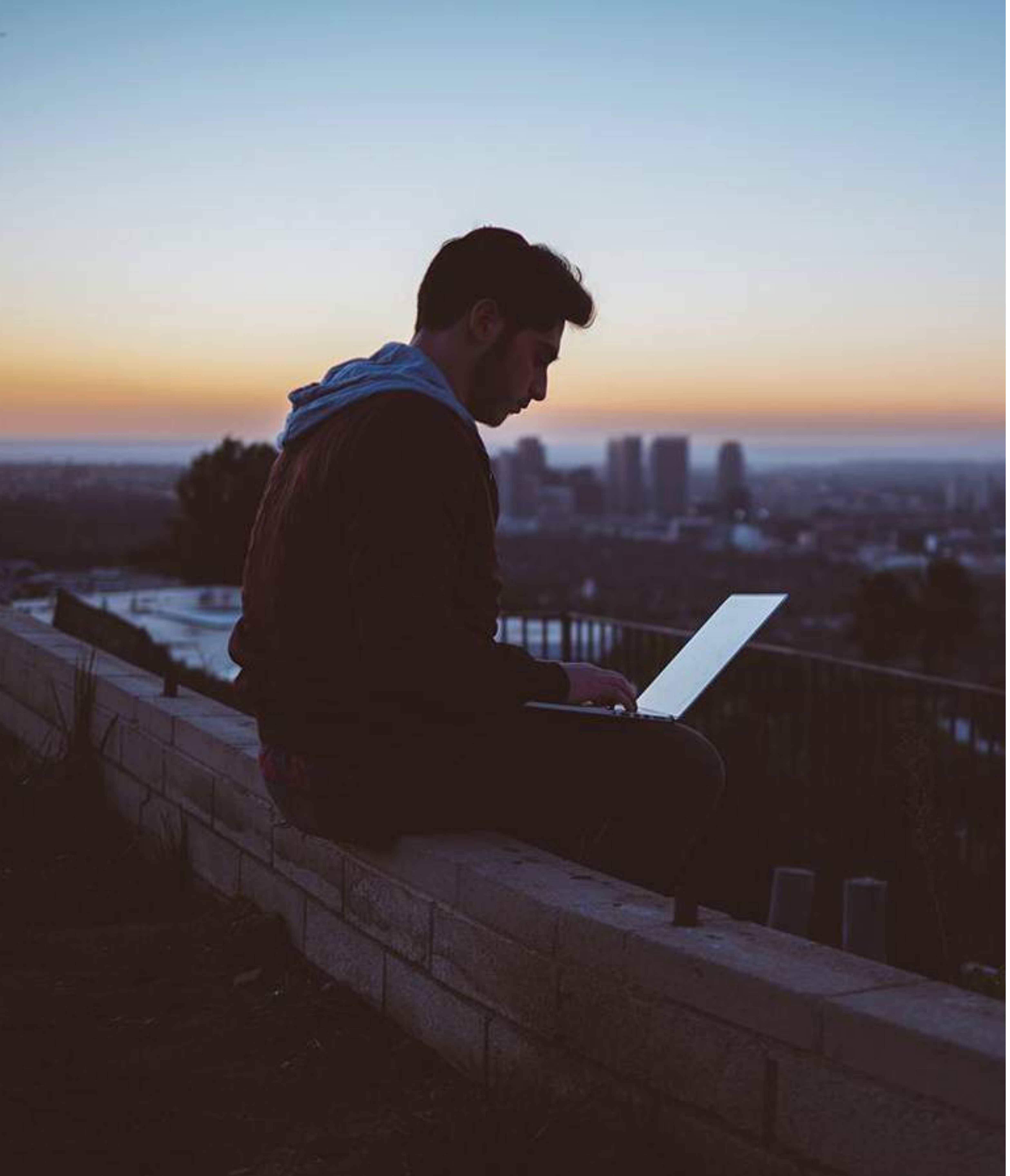
Build secrets and SSH forwarding in Docker 18.09

Tõnis Tiigi Follow
Nov 8 · 6 min read



3. docker build --ssh=default . (docker)

```
# docker build --ssh=default .
[+] Building 9.0s (8/9)
 => [internal] load build definition from Dockerfile                        0.0s
 => => transferring dockerfile: 386B                                        0.0s
 => [internal] load .dockerignore                                           0.0s
 => => transferring context: 2B                                             0.0s
 => resolve image config for docker.io/docker/dockerfile-upstream:experimenta 1.0s
 => CACHED docker-image://docker.io/docker/dockerfile-upstream:experimental@s 0.0s
 => [internal] load metadata for docker.io/library/alpine:latest            0.3s
 => [1/4] FROM docker.io/library/alpine@sha256:621c2f39f8133acb8e64023a94dbdf 0.0s
 => => resolve docker.io/library/alpine@sha256:621c2f39f8133acb8e64023a94dbdf 0.0s
 => => sha256:621c2f39f8133acb8e64023a94dbdf0d5ca81896102b9e5 2.03kB / 2.03kB 0.0s
 => => sha256:02892826401a9d18f0ea01f8a2f35d328ef039db4e1edcc45c6 528B / 528B 0.0s
 => => sha256:196d12cf6ab19273823e700516e98eb1910b03b17840f9d 1.51kB / 1.51kB 0.0s
 => [2/4] RUN apk add --no-cache openssh-client git                         2.6s
 => [3/4] RUN mkdir -p -m 0600 ~/.ssh && ssh-keyscan github.com >> ~/.ssh/kno 2.7s
 => [4/4] RUN --mount=type=ssh ssh git@github.com 2>&1 | grep "Hi tonistiigi" 1.3s
▌
```



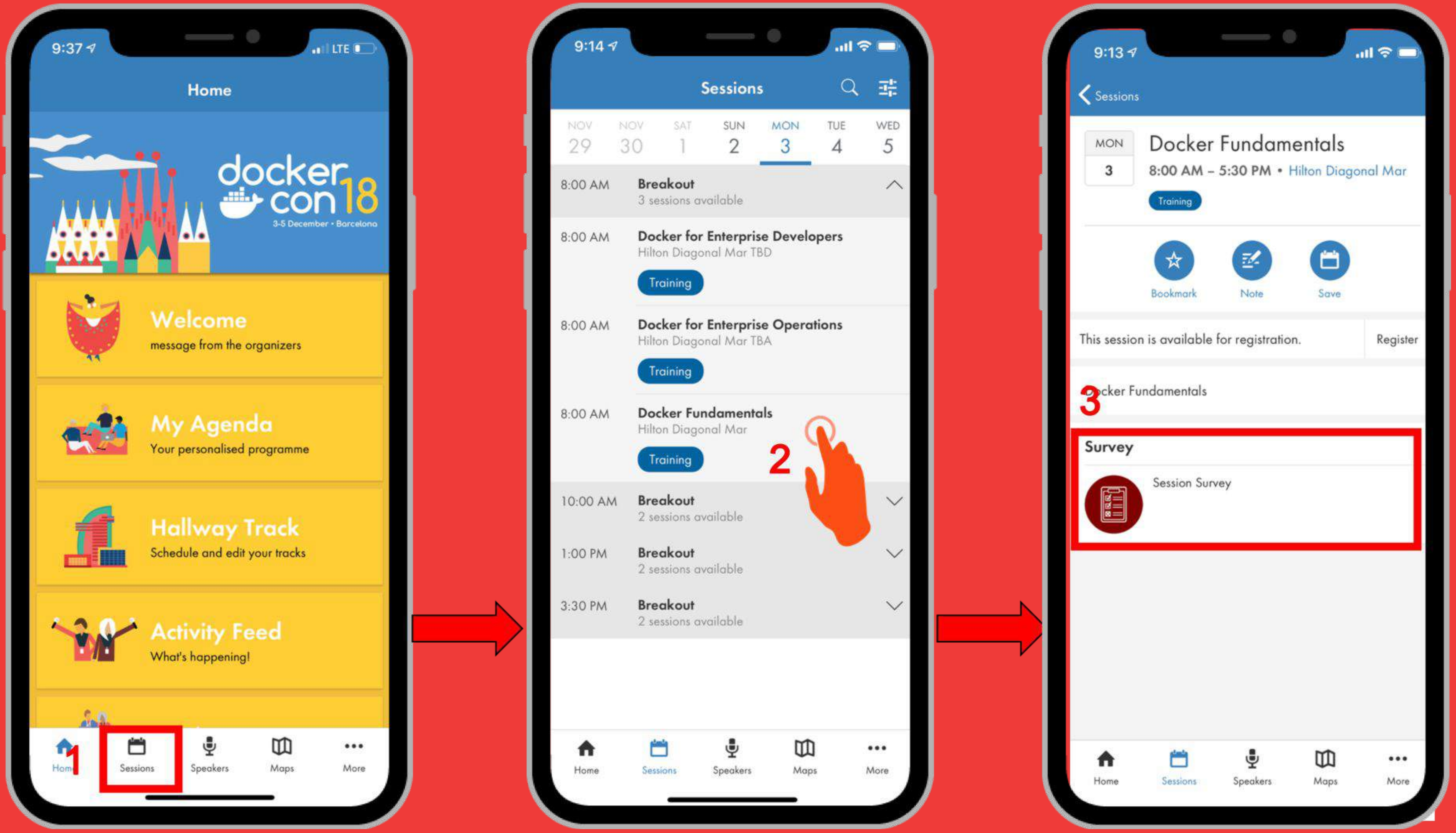
Thank you!

- Multi-stage, multi-stage, multi-stage
- Enable BuildKit
- *Supercharged Docker Build with BuildKit* in **BlackBelt** session on **Wednesday at 12pm**

Take A Breakout Survey

Access your session and/or workshop surveys for the conference at any time by tapping the Sessions link on the navigation menu or block on the home screen.

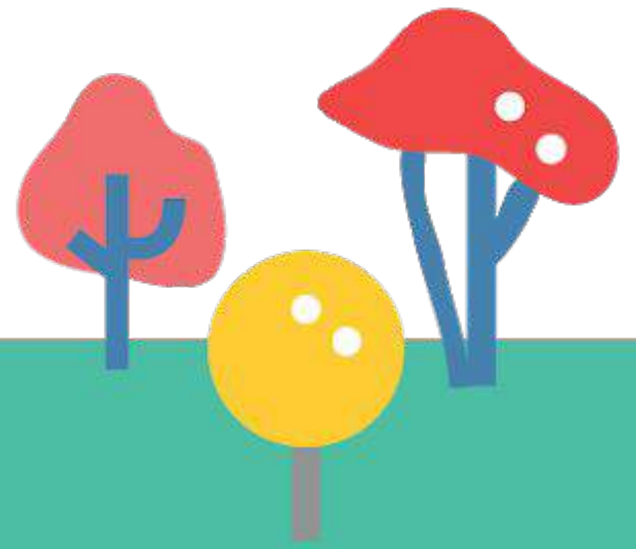
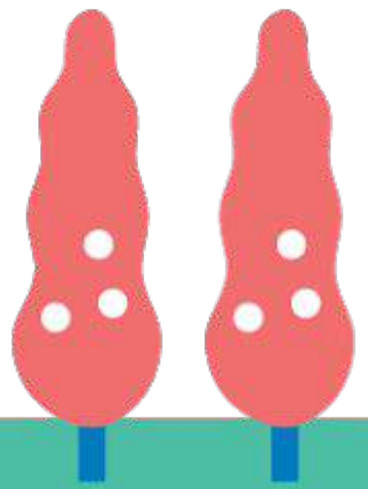
Find the session/workshop you attended and tap on it to view the session details. On this page, you will find a link to the survey.



dockercon18 EUROPE

Run as an unprivileged user

```
FROM openjdk:8-jre-alpine
RUN addgroup -g 50 -S appuser \
 && adduser -D -S -h /app -s /sbin/nologin \
    -u 1000 -G appuser appuser
USER appuser:appuser
COPY app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Run as an unprivileged user

```
FROM openjdk:8u181-jre-alpine
RUN addgroup -g 50 -S appuser \
 && adduser -D -S -h /app -s /sbin/nologin \
    -u 1000 -G appuser appuser
USER appuser:appuser
COPY app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Build secrets

```
# syntax=docker/dockerfile:1.0-experimental

FROM ...
RUN --mount=type=secret,id=mysecret,required ...
```

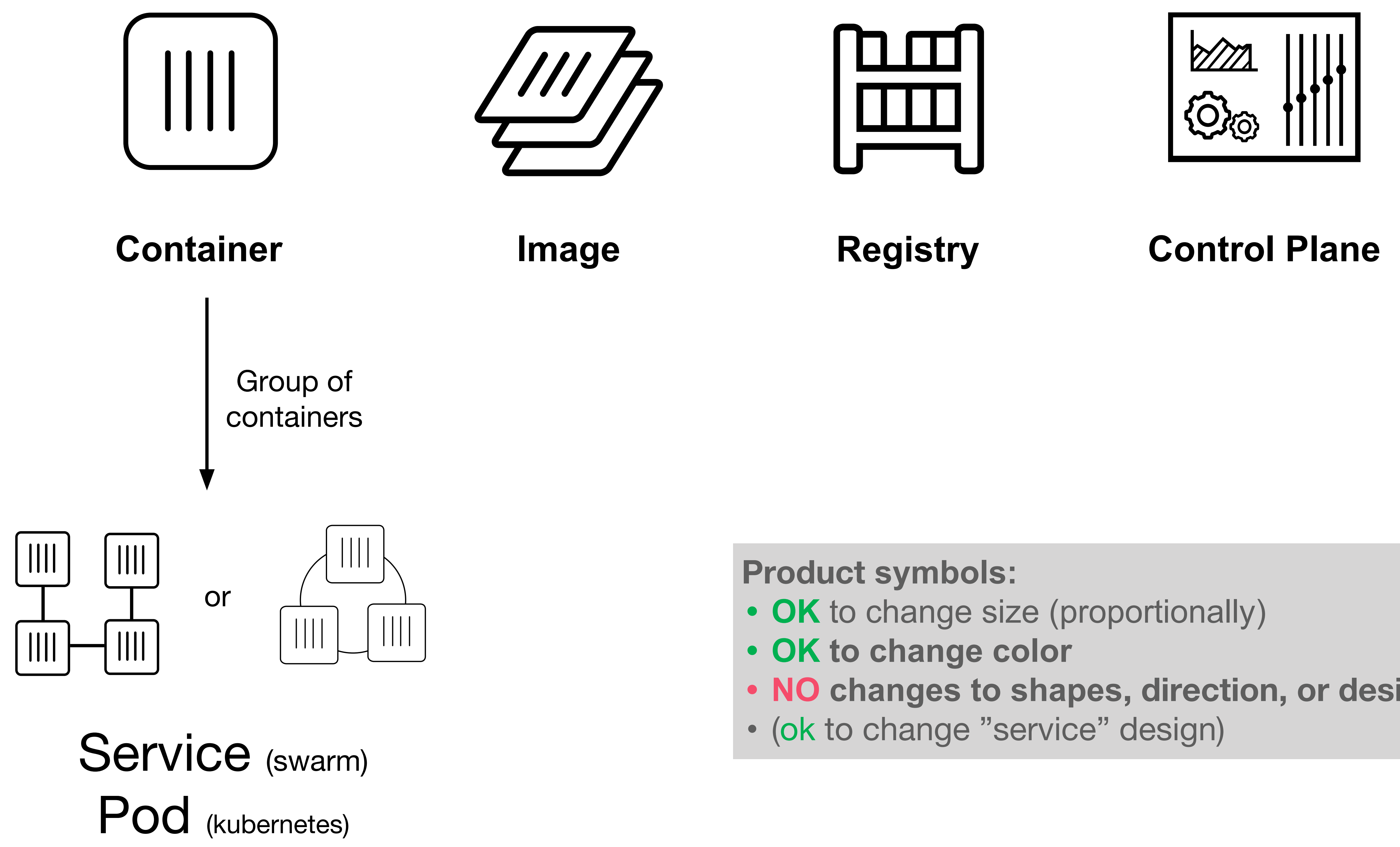
```
$ docker build --secret id=mysecret,src=/local/secret .
```

SSH

```
# syntax=docker/dockerfile:1.0-experimental

FROM ...
RUN --mount=type=ssh git clone git@github.com:myorg/myproject.git
```

```
$ docker build --ssh default
```

Docker Product / Feature Icons



Product symbols:

- **OK** to change size (proportionally)
- **OK to change color**
- **NO changes to shapes, direction, or design**
- (ok to change ”service” design)

Icons

Computer, PC, terminal, laptop, device

Develop dev

Mobile watch

Edge Device

Server, data center

Storage database

Cloud

Globe, location

Layer, vm

VM

Network

Metrics, alert, dashboard Monitor, logging, operations configure

Repair tune

Relationship, hierarchy, process, integration, arrows, cycle

Check

CI /

dockercon18 EUROPE

Security, secure, Scan, key, sign, encrypt

firewall

Clocks, speed, time

Process, relationship, hierarchy, cycle

integrate

CI /

trust

Chain, broken

Calendar, date

People

executive

architect

practitioner

developer

Operator, support

Generic male

Generic female

Generic speaker

group

People Misc

Hands - Shake
Agreement - button

docker con 18 EUROPE

- **20+ Websites for Incredible Free Stock Photos**
  - https://mymodernmet.com/best-free-stock-photography-websites/
  - Includes sites focusing on food, nature, places, vintage, humorous/whimsical as well as general photo sites
- **21 Amazing Sites With Breathtaking Free Stock Photos**
  - https://blog.snappa.com/free-stock-photos/

Generic Block Diagrams
Calls to Action
Summary Groups