KARSTEN SILZ

15 MAY 2020

HTTPS://BPF.LI

# GOOGLE JIB: SMALLER & FASTER DOCKER IMAGES FOR JAVA APPLICATIONS

# SLIDES & SOURCE CODE

HTTPS://BPF.LI/PGJD

DO YOU CREATE JAVA DOCKER IMAGES BY COPYING JAR FILES INTO THE IMAGE?

DO YOU WANT TO SAVE TIME & NETWORK BANDWIDTH?

# CAN YOU CHANGE YOUR GRADLE/MAVEN BUILD?

# GOOGLE JIB

## SMALLER JAVA DOCKER IMAGES
## FASTER DOCKER PUSHES

# DOCKERFILES & LAYERS

Dockerfile create Docker Images

Docker Images have multiple layers

Most Dockerfile lines create new layer

Docker stores & caches layers individually, both locally and when pushing to/pulling from Docker repository

When one layer changes, then all following layers also get rebuilt

PUT THINGS THAT CHANGE
MOST OFTEN AT THE
END OF DOCKERFILE

# SAMPLE DOCKERFILE

```
FROM adoptopenjdk/openjdk11-openj9:x86_64-
debianslim-jre-11.0.7_10_openj9-0.20.0
```
LAYER 1

```
RUN mkdir -p /usr/app
```
LAYER 2

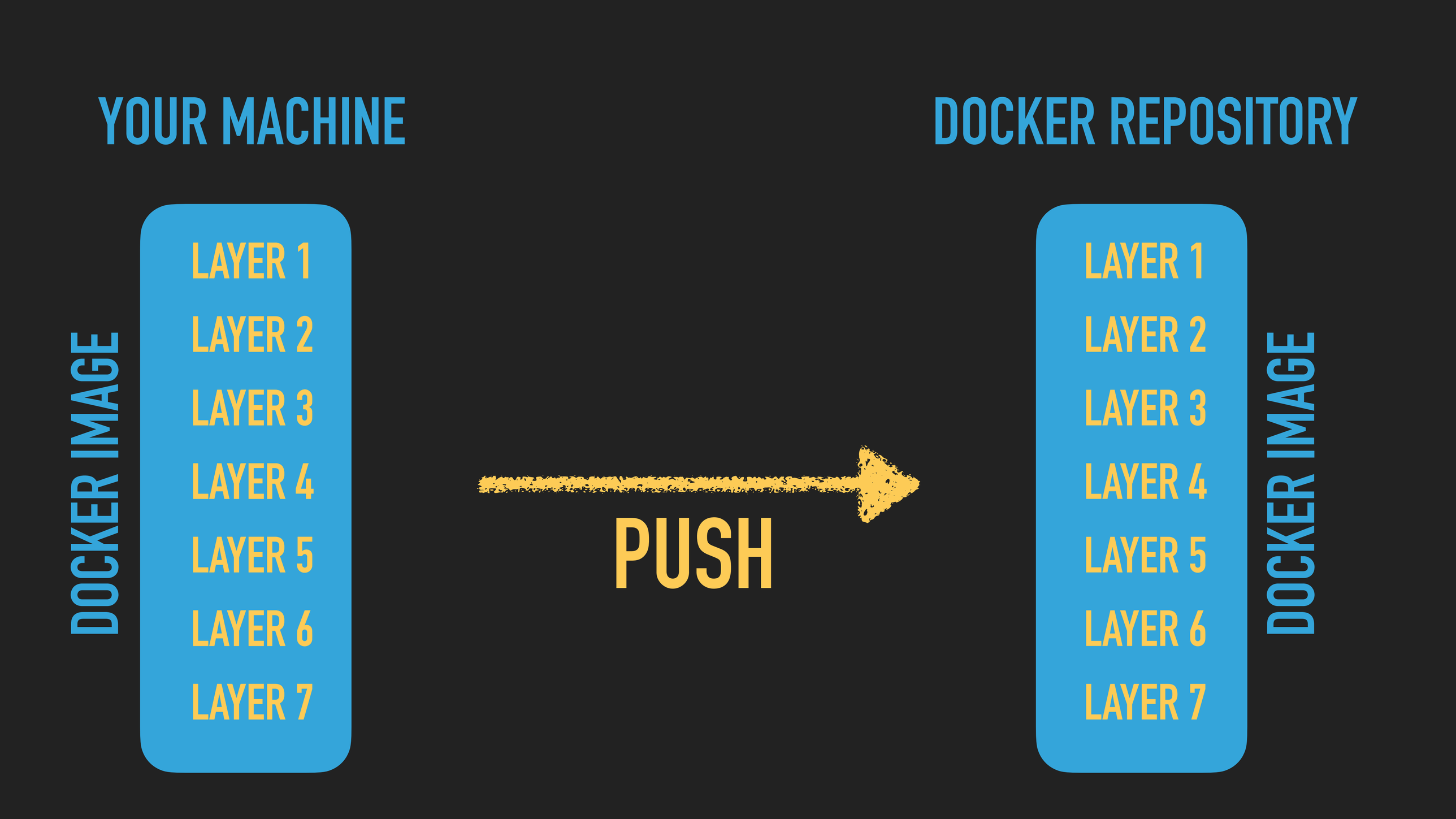
```
WORKDIR /usr/app
```
LAYER 3

```
ENV JHIPSTER_SLEEP=0
```
LAYER 4

```
COPY entrypoint.sh /usr/app/entrypoint.sh
```
LAYER 5

```
COPY simple-shop-1.0.0.jar /usr/app/simple-shop.jar
```
LAYER 6

```
ENTRYPOINT ["sh", "-c", "chmod +x /usr/app/
entrypoint.sh && cd /usr/app && ./entrypoint.sh"]
```
LAYER 7


YOUR MACHINE
DOCKER REPOSITORY
DOCKER IMAGE
LAYER 1
LAYER 2
LAYER 3
LAYER 4
LAYER 5
LAYER 6
LAYER 7
PUSH
LAYER 1
LAYER 2
LAYER 3
LAYER 4
LAYER 5
LAYER 6
LAYER 7
DOCKER IMAGE

# ALL 7 LAYERS GET PUSHED

# YOU CHANGE YOUR APPLICATION

YOU CHANGE ONE JAVA CLASS

= ONE CLASS FILE & RESOURCES
(BUILD INFO, WEB APP) CHANGE

= NEW JAR FILE

```
FROM adoptopenjdk/openjdk11-openj9:x86_64-
debianslim-jre-11.0.7_10_openj9-0.20.0
```
LAYER 1

```
RUN mkdir -p /usr/app
```
LAYER 2

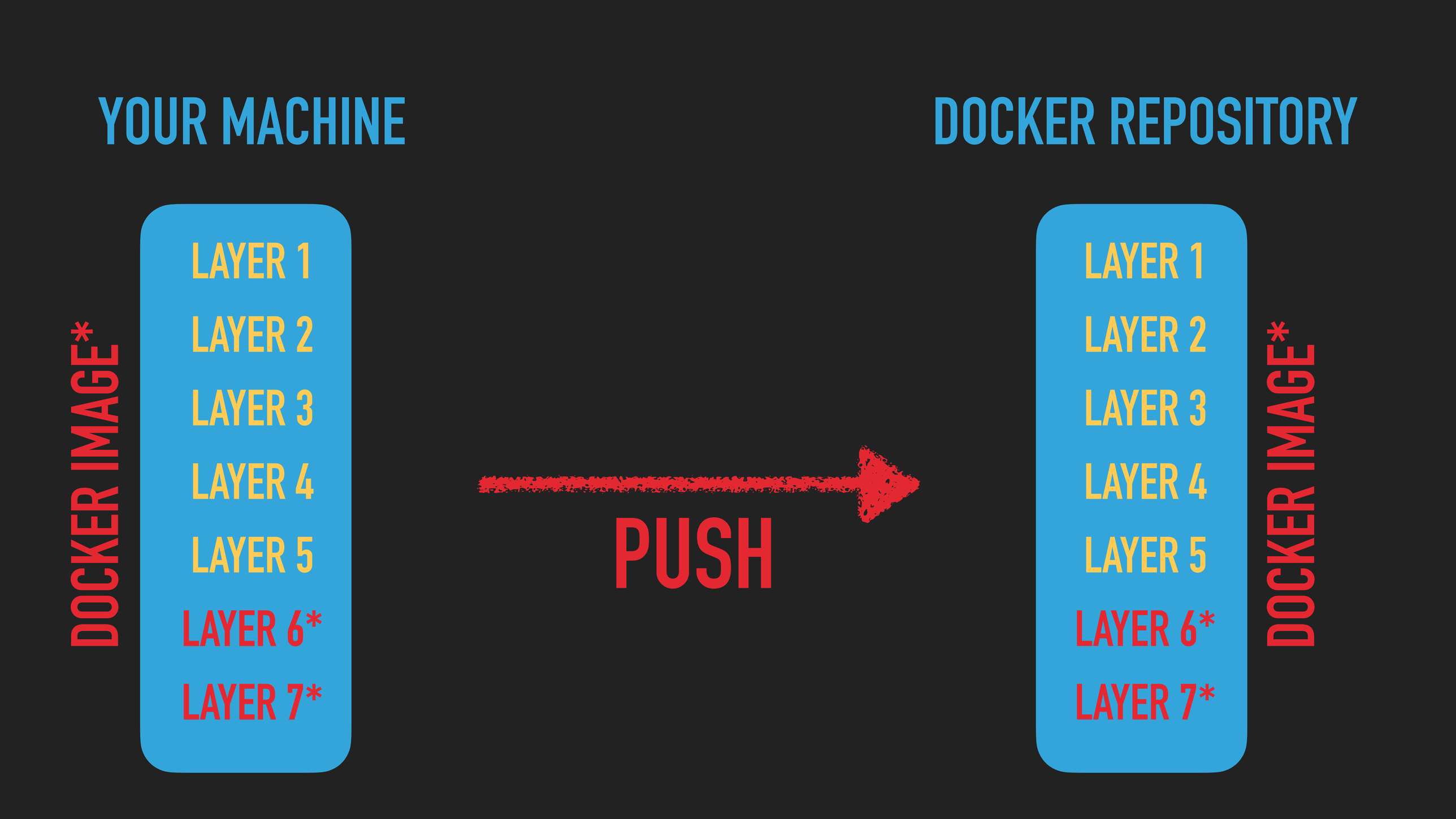
```
WORKDIR /usr/app
```
LAYER 3

```
ENV JHIPSTER_SLEEP=0
```
LAYER 4

```
COPY entrypoint.sh /usr/app/entrypoint.sh
```
LAYER 5

```
COPY simple-shop-1.0.1.jar /usr/app/simple-shop.jar
```
LAYER 6*

```
ENTRYPOINT ["sh", "-c", "chmod +x /usr/app/
entrypoint.sh && cd /usr/app && ./entrypoint.sh"]
```
LAYER 7*


YOUR MACHINE
DOCKER REPOSITORY
DOCKER IMAGE*
LAYER 1
LAYER 2
LAYER 3
LAYER 4
LAYER 5
LAYER 6*
LAYER 7*
PUSH
LAYER 1
LAYER 2
LAYER 3
LAYER 4
LAYER 5
LAYER 6*
LAYER 7*
DOCKER IMAGE*

# 2 CHANGED LAYERS GET PUSHED

EVERY CHANGE:
NEW JAR FILE = NEW LAYER

PUSHING THAT NEW LAYER
WASTES TIME & BANDWIDTH

# GOOGLE JIB
# UNPACKS JAR

Open source Docker image build tool by Google

Gradle & Maven plugins, Java library

Jib configuration instead of Dockerfile

Minimizes Docker image layer changes

Bonus: Build Docker image without Docker daemon

# SAMPLE DOCKER IMAGE

`adoptopenjdk/openjdk11-openj9:x86_64-debianslim-jre-11.0.7_10_openj9-0.20.0` — LAYER 1

`/app/libs` — *Dependency JAR files* — LAYER 2

`/app/resources` — *Resource files* — LAYER 3

`/app/classes` — *Your classes* — LAYER 4

`/` — *Extra files* — LAYER 5

# ALL 5 LAYERS GET PUSHED

# YOU CHANGE YOUR APPLICATION

YOU CHANGE ONE JAVA CLASS

= ONE CLASS FILE & RESOURCES
(BUILD INFO, WEB APP) CHANGE

= 2 DIRECTORIES CHANGE

```
FROM adoptopenjdk/openjdk11-openj9:x86_64-
debianslim-jre-11.0.7_10_openj9-0.20.0
```

LAYER 1

`/app/libs` *Dependency JAR files*

LAYER 2

`/app/resources` *Resource files*

LAYER 3*

`/app/classes` *Your classes*

LAYER 4*

`/` *Extra files*

LAYER 5*

3 CHANGED LAYERS GET PUSHED

BUT THEY ARE MUCH SMALLER!

# HOW MUCH DO YOU SAVE WITH JIB?

# SMALL SPRING BOOT ANGULAR WEB APPLICATION

| Data to Push: | Class or Resource Changes | Dependency JAR Changes |
|---|---|---|
| Dockerfile | 61.4 MB | 61.4 MB |
| Jib | 4.7 MB | 64.5 MB |
| Jib savings | 92% | -5% |

# PRE-PRODUCTION SPRING BOOT ANGULAR WEB APPLICATION

| Data to Push: | Class or Resource Changes | Dependency JAR Changes |
|---|---|---|
| Dockerfile | 124 MB | 124 MB |
| Jib | 7.5 MB | 128.5 MB |
| Jib savings | 94% | -4% |

DOCKER USUALLY PUSHES
90%+ LESS DATA WITH JIB

# WHY?

CHANGED LAYERS
ARE SMALLER IN JIB

= DOCKER PUSHES FASTER

# GOOGLE JIB

## SMALLER JAVA DOCKER IMAGES
## FASTER DOCKER PUSHES

# SLIDES & SOURCE CODE

HTTPS://BPF.LI/PGJD