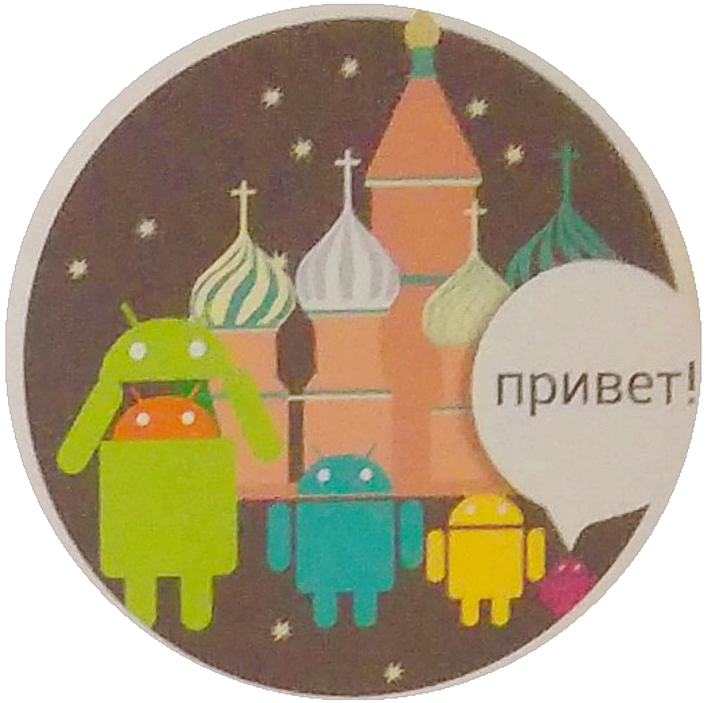
Android meets Docker

Jing Li

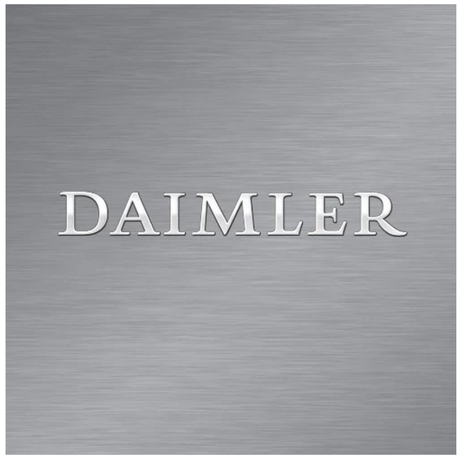
mobius


Jing
GoPro
MIND THE GAP
привет!
你好!
CODE THE ROAD
Hello!
¡Hola!
I amsterdam
Hi!
Bonjour!
Hello!
01
CALIFORNIA REPUBLIC
LONDON
B. awesome


привет!

mobius

> 50 cities in Europe



mobius

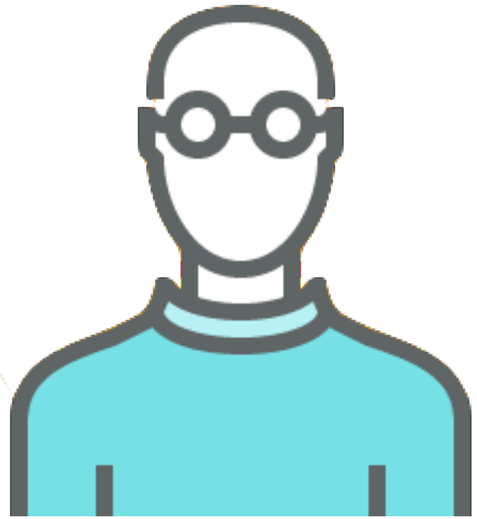
Developer Story

```
git: command not found
```

```
gradle: command not found
```

```
SDK location not found. Define location with sdk.dir in the local.properties
file or with an ANDROID_HOME environment variable.
```

```
No Java runtime present, requesting install.
```



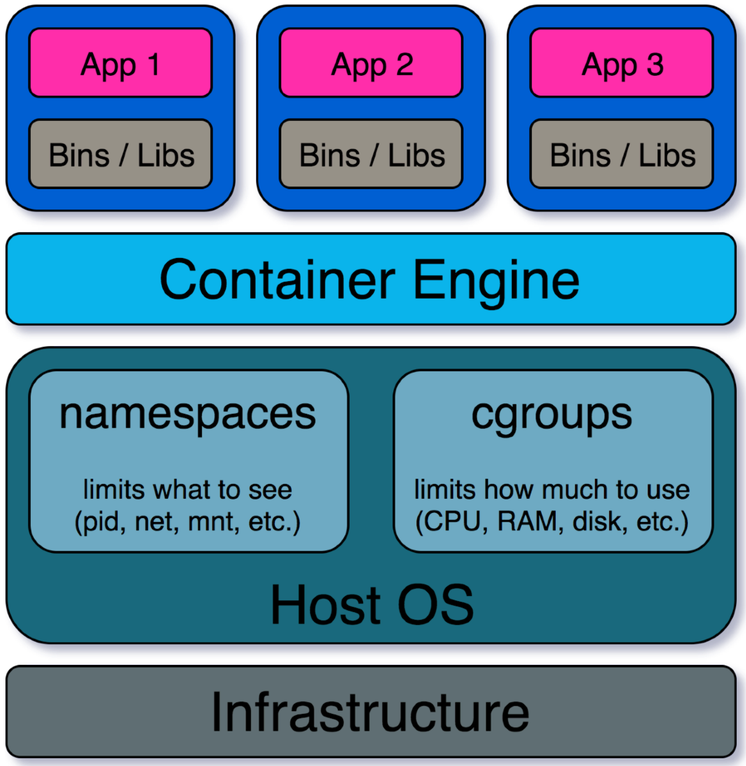
mobius

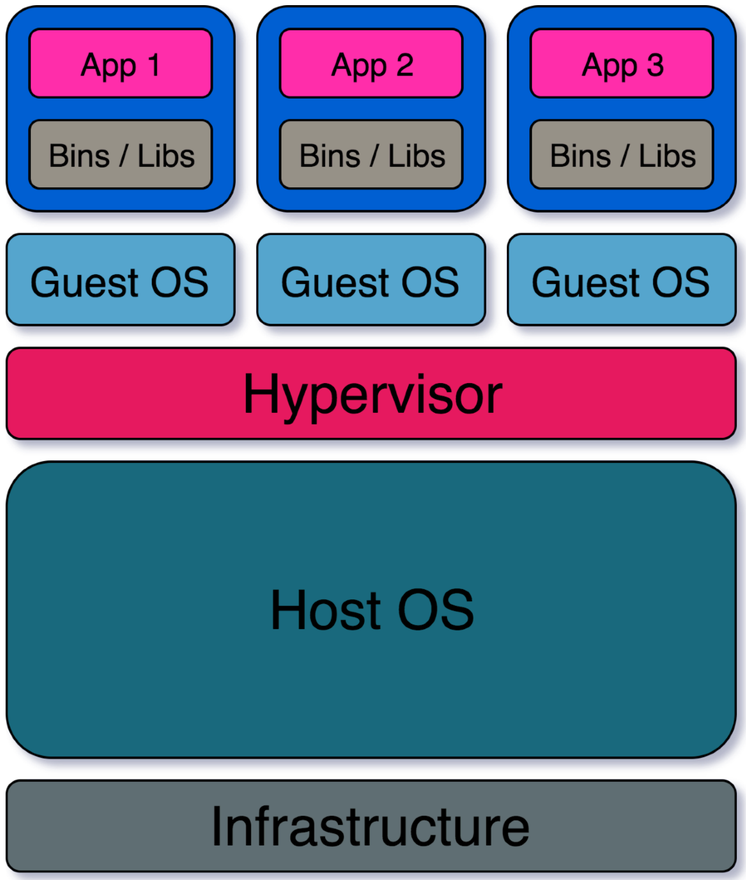
Pain in the Admin

provision ∞ machines (e.g. mobile CI)

mobius

Containerization vs Virtualization





mobius

Why Docker?

	Docker	Vagrant
Resource Isolation	Low	Extreme
Supported OS	Linux	*
Boot / Stop Time	Seconds	Minutes
Size	MB	GB

mobius

Docker for Android?

Infer - A tool to detect bugs in Java and ... code.

Run Android SDK u[illegible]te in Docker container

AUFS doesn't support hardlink move operations.

mobius

Docker ABC

- image vs container
- build vs pull
- Let's go!

mobius

Dockerfile reference

- FROM - base image
- RUN - execute commands in a new layer
- ENV - set the environment variable
- ADD - copy files to image
- EXPOSE - listen on the specified network port
- CMD - provide defaults for an executing container

mobius

Dockerfile

```
FROM ubuntu:16.04

# support multiarch: i386 architecture
# install Java
# install essential tools
# install Qt
RUN dpkg --add-architecture i386 && \
    apt-get update -y && \
    apt-get install -y libncurses5:i386 libc6:i386 libstdc++6:i386 lib32gcc1
lib32ncurses5 lib32z1 zlib1g:i386 && \
    apt-get install -y --no-install-recommends openjdk-8-jdk && \
    apt-get install -y git wget zip && \
    apt-get install -y qt5-default
```

Dockerfile

```
# download and install Gradle
ENV GRADLE_VERSION 4.2.1
RUN cd /opt && \
    wget -q https://services.gradle.org/distributions/gradle-${GRADLE_VERSION}-bin.zip && \
    unzip gradle*.zip && \
    ls -d */ | sed 's/\/*$//g' | xargs -I{} mv {} gradle && \
    rm gradle*.zip

# download and install Kotlin compiler
ENV KOTLIN_VERSION 1.1.51
RUN cd /opt && \
    wget -q https://github.com/JetBrains/kotlin/releases/download/v${KOTLIN_VERSION}/kotlin-
compiler-${KOTLIN_VERSION}.zip && \
    unzip *kotlin*.zip && \
    rm *kotlin*.zip
```

Dockerfile

```
# download and install Android SDK
ENV ANDROID_SDK_VERSION 3859397
RUN mkdir -p /opt/android-sdk && cd /opt/android-sdk && \
    wget -q https://dl.google.com/android/repository/sdk-tools-linux-${ANDROID_SDK_VERSION}.zip && \
    unzip *tools*linux*.zip && \
    rm *tools*linux*.zip

# set the environment variables
ENV JAVA_HOME /usr/lib/jvm/java-8-openjdk-amd64
ENV GRADLE_HOME /opt/gradle
ENV KOTLIN_HOME /opt/kotlinc
ENV ANDROID_HOME /opt/android-sdk
ENV PATH ${PATH}:${GRADLE_HOME}/bin:${KOTLIN_HOME}/bin:${ANDROID_HOME}/tools:${ANDROID_HOME}/platform-tools:$
{ANDROID_HOME}/tools/bin:${ANDROID_HOME}/emulator
ENV _JAVA_OPTIONS -XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap
# WORKAROUND: for issue https://issuetracker.google.com/issues/37137213
ENV LD_LIBRARY_PATH ${ANDROID_HOME}/emulator/lib64:${ANDROID_HOME}/emulator/lib64/qt/lib
```

Dockerfile

```
# accept the license agreements of the SDK components
ADD license_accepter.sh /opt/
RUN /opt/license_accepter.sh $ANDROID_HOME

# install and configure SSH server
ADD banner.net /etc/
ADD authorized_keys /tmp/
EXPOSE 22
RUN apt-get update -y && \
    apt-get install -y openssh-server supervisor locales && \
    ...
ADD supervisord.conf /etc/supervisor/conf.d/
CMD ["/usr/bin/supervisord"]
```

mobius

Dockerfile Best Practices

- **Single responsibility**

  `use compose for orchestration`

- **Minimize the number of layers**

  `layer = intermediate image, supports caching`

- **Size matters, use what you have**

  `Install & un- by separated steps will ⇧ image size`

- **Readability**

mobius

Solution for SDK update

Mount SDK volume in container

- Minimal
- Flexible
- Persist
- Share

mobius

Different Approaches

- Mount SDK volume
- BTRFS storage driver
- One image per Android API level

NFS

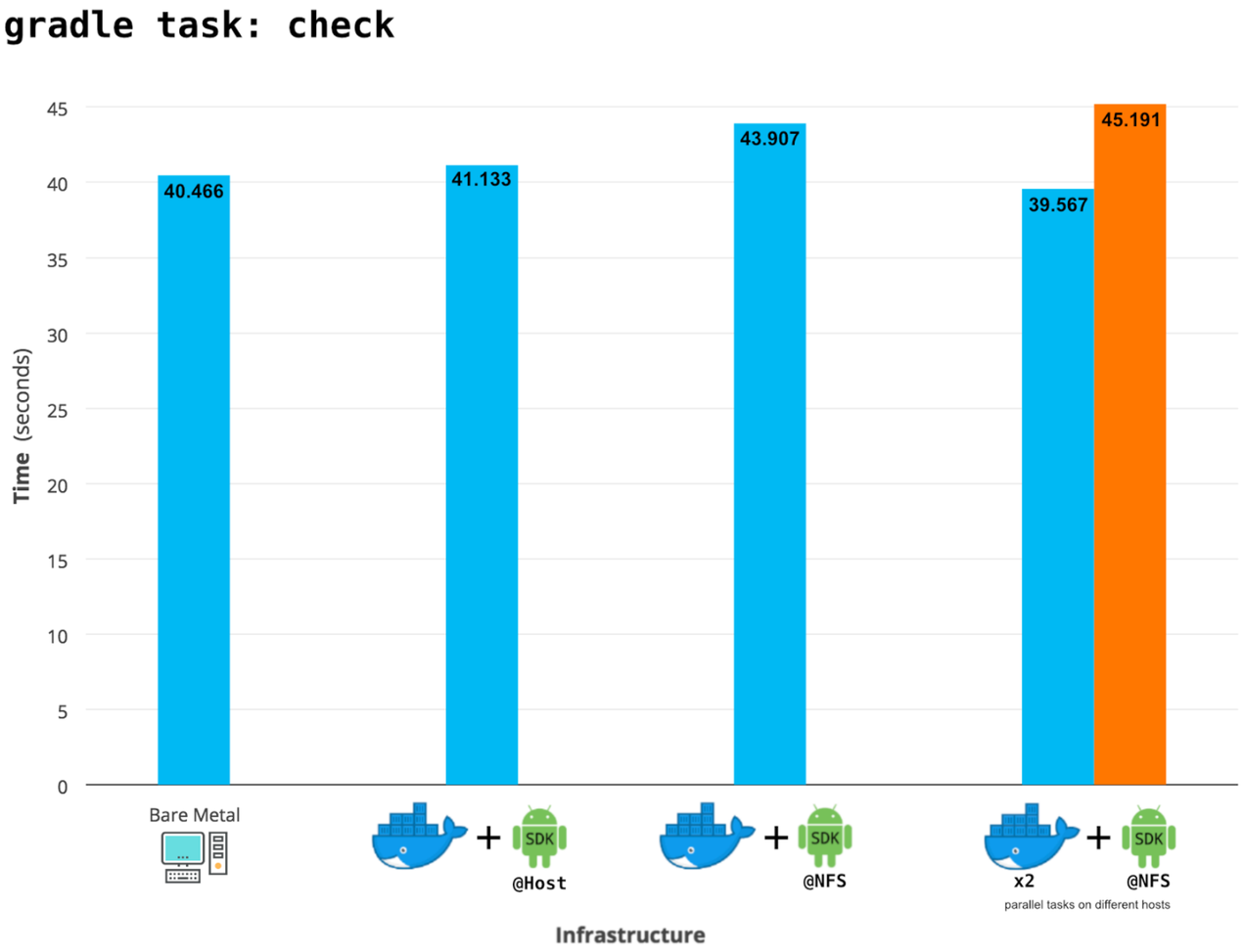
✓ In one place

? Performance

⚠ No concurrent writing

mobius

Benchmark


gradle task: check
Time (seconds)
0
5
10
15
20
25
30
35
40
45
40.466
41.133
43.907
39.567
45.191
Bare Metal
@Host
@NFS
x2
@NFS
parallel tasks on different hosts
Infrastructure

Performance Comparison

2 build types, 107 unit tests (x2 = 214), 2 UI tests `./gradlew clean check :demo:connectedAndroidTest`	
On-premises Docker (w/ cached AndroidSDK)	6 mins 48.0 secs
Other online CI solutions (w/o optimization)	> 10 mins

mobius

Android Devices

- ARM emulator
- x86 emulator (requires KVM)
- USB (needs privileged mode, ⛔ for macOS)
- Wifi
- Genymotion Cloud

mobius

Performance of 📱

2 UI tests	
ARM emulator @🐳	2 mins 4.615 secs
x86 emulator @🐳 (on Linux Host)	23.497 secs
Genymotion	25.335 secs

mobius

Out Of Memory

```
Killed
```

```
Process 'Gradle Test Executor 1' finished with non-zero exit value 137
Gradle build daemon disappeared unexpectedly (it may have been killed or may have crashed)
```

```
java.lang.OutOfMemoryError
```

```
* What went wrong:
Execution failed for task ':docker:testDebugUnitTest'.
> Process 'Gradle Test Executor 1' finished with non-zero exit value 1
```

mobius

Memory Matters

JVM is not container aware

_JAVA_OPTIONS		-XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap
Exit Code	137 = 128 + 9 = SIGKILL = Killed	1 = SIGHUP = Hangup
Comment	killed by the kernel OOM killer	JVM terminates the program and exits

SSH

- **Mind your language**

```
locale-gen en en_US en_US.UTF-8
```

- **Set up env**

```
/root/.ssh/environment
```

- **Authorization**

```
/root/.ssh/authorized_keys
```

Jenkins Env Var

- **Global level**

  Configure System -> Global properties -> Environment variables

- **Node level**

  Manage Nodes -> Configure Node -> Node Properties -> Environment variables

- **Job level**

  Configure Job -> Build -> Build Step -> Execute shell
  Plugin: Environment Injector -> Inject variables to the build process / as a build step

Can we get any better?

Gradle distributions

mobius

Gradle distributions mirror server

- From: `gradle/wrapper/gradle-wrapper.properties`
- To: `~/.gradle/wrapper/dists`
- SSL cert - needs to be trusted by Java keystore
- /etc/hosts

```
#Sat Jun 17 13:32:16 CEST 2017
distributionBase=GRADLE_USER_HOME
distributionPath=wrapper/dists
zipStoreBase=GRADLE_USER_HOME
zipStorePath=wrapper/dists
distributionUrl=https\://services.gradle.org/distributions/gradle-3.3-all.zip
```

mobius

And better?

Gradle caches

- Don't waste time downloading dependencies
- `~/.gradle/caches/`

Reveal Machine-dependent Problem

- Encoding problem

  ```
  expected:<Hall[]chen> but was:<Hall[ö]chen>
  ```

- Hard coded Timezone in test

  ```
  expected: 2099-12-31T00:00:00.0000+0200
  but was : 2099-12-31T00:00:00.0000+0000
  ```

- File & Path

  ```
  File#listFiles() -> File[] (sort order depends on OS)
  ```

mobius

What Else for Mobile

- Integration Test
  - Prod / Test server
    - unreliable network
    - complex setup
  - Docker
    - out of the box for mobile dev
- Something else
  - Serverless (Kotlin, Swift) - reuse  code

mobius

Hiring

👍 Offices in **Hamburg**, **Berlin**, **Barcelona** and etc.

👍 Visa sponsor, offering relocation

- Android Developer
- iOS Developer
- Backend / Frontend / Data / * Engineers

https://de.mytaxi.com/jobs

mobius

спасибо!

thyrlian

thyrlian

Docker Image:
https://github.com/thyrlian/AndroidSDK

mobius