KubeCon | CloudNativeCon

Europe 2018

Building Docker Images without Docker

Matt Rickard, Software Engineer, Google

@mattrickard

KubeCon | CloudNativeCon

Europe 2018

How to create compressed tarballs in the most complicated ways imaginable

Matt Rickard, Software Engineer, Google
@mattrickard

Agenda

- Why do we care?
- What did we try to do?
- What problems did we run into?
- What do we do now?

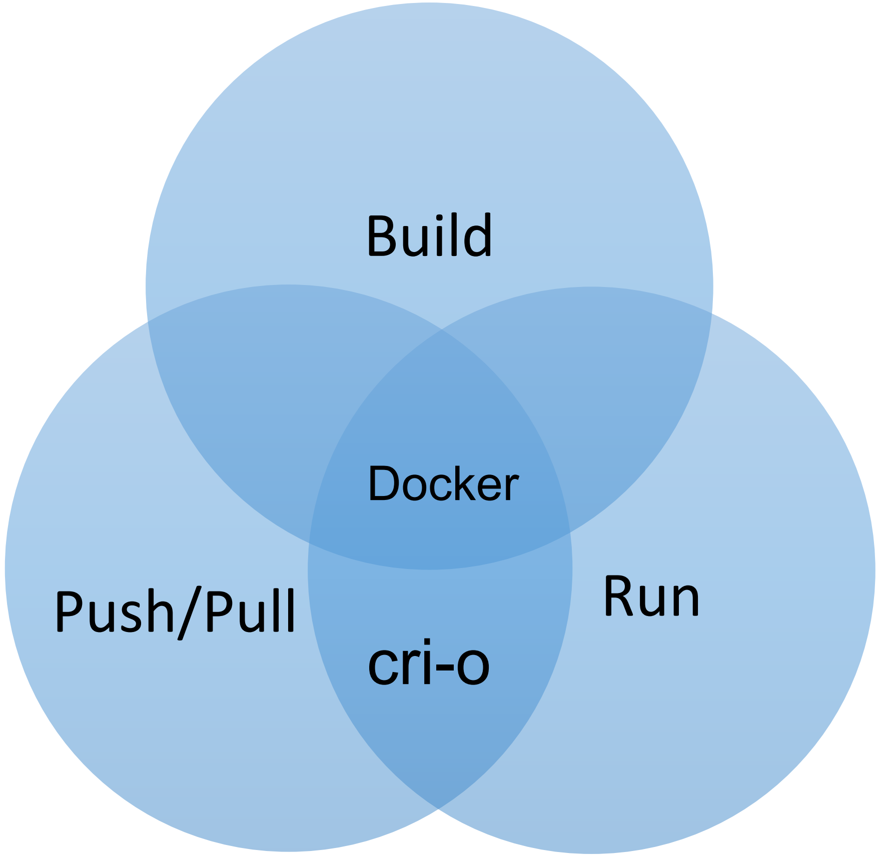
Who am I?

- Matt Rickard
- Software Engineer at Google San Francisco
- Open Source Software
  - Containers
  - Kubernetes Developer Experience
    - kubernetes/minikube
    - GoogleContainerTools/**
      - skaffold, distroless, kaniko, container-diff
- We’re hiring!
  - mrick@google.com

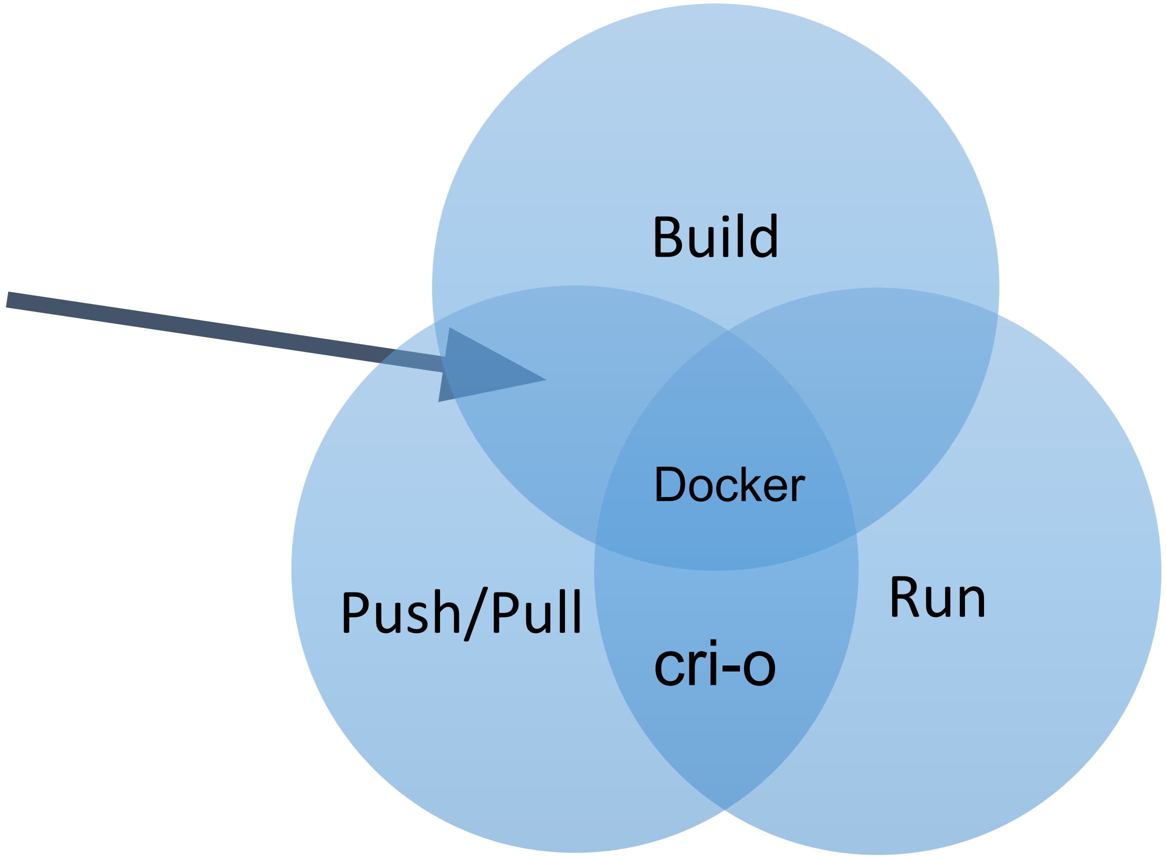
Why do we care?

- Separation of concerns
  - build, pull/push, login, run
- Security
- Reuse and consolidate infrastructure
  - on-cluster builds
- Reproducibility
- Minimal images
- Control over images
  - Layers
  - Contents

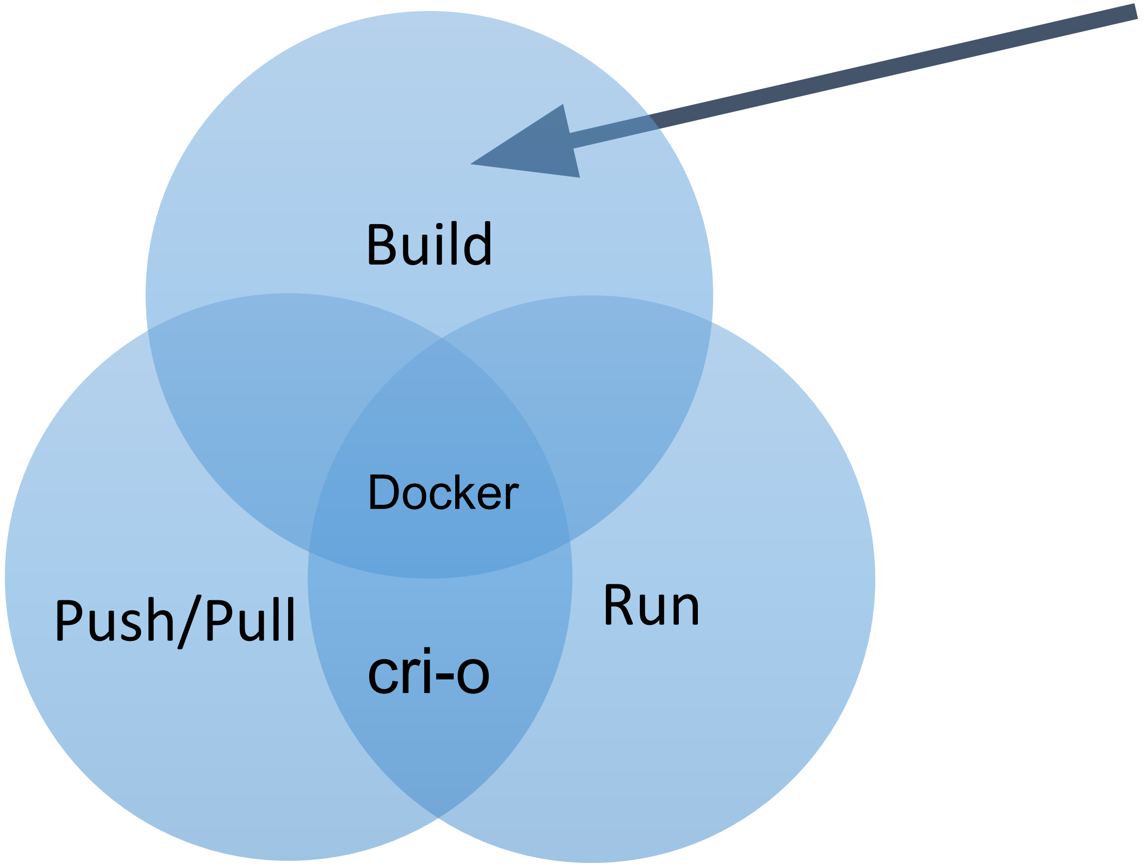
Separation of Concerns



Aim for Build+(Push/Pull)



Build Only is Ideal



Build

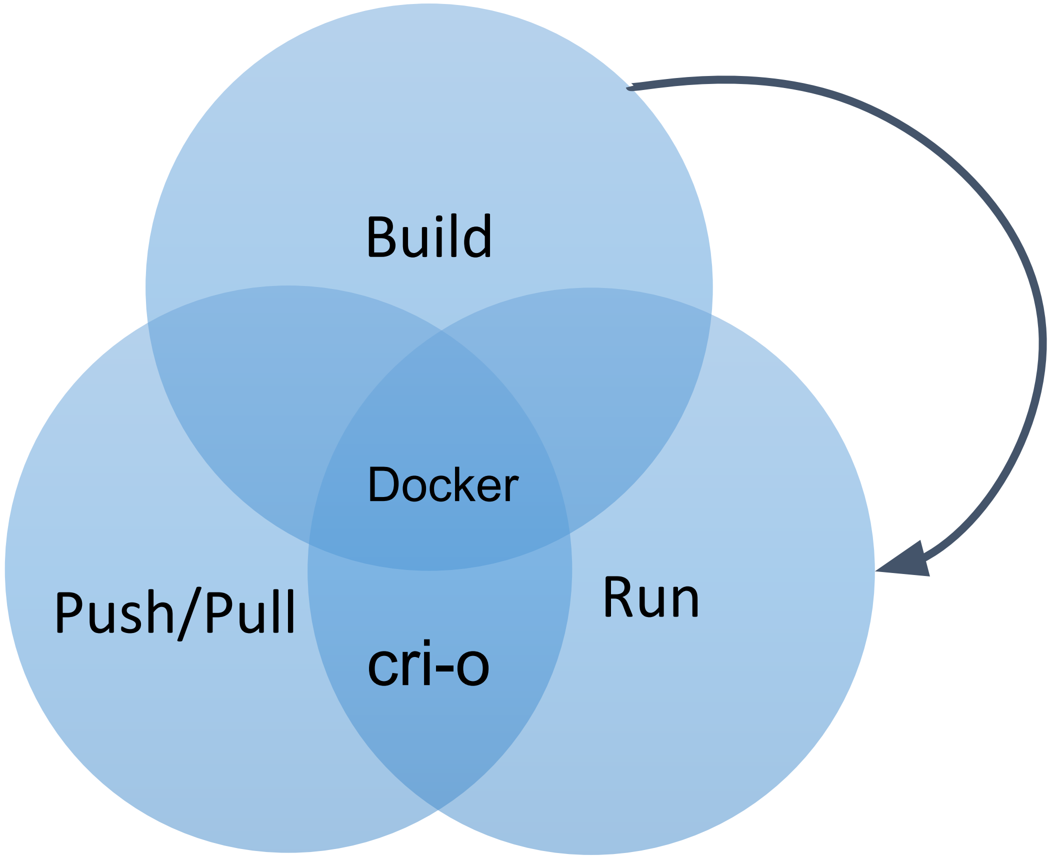
Docker

Push/Pull

cri-o

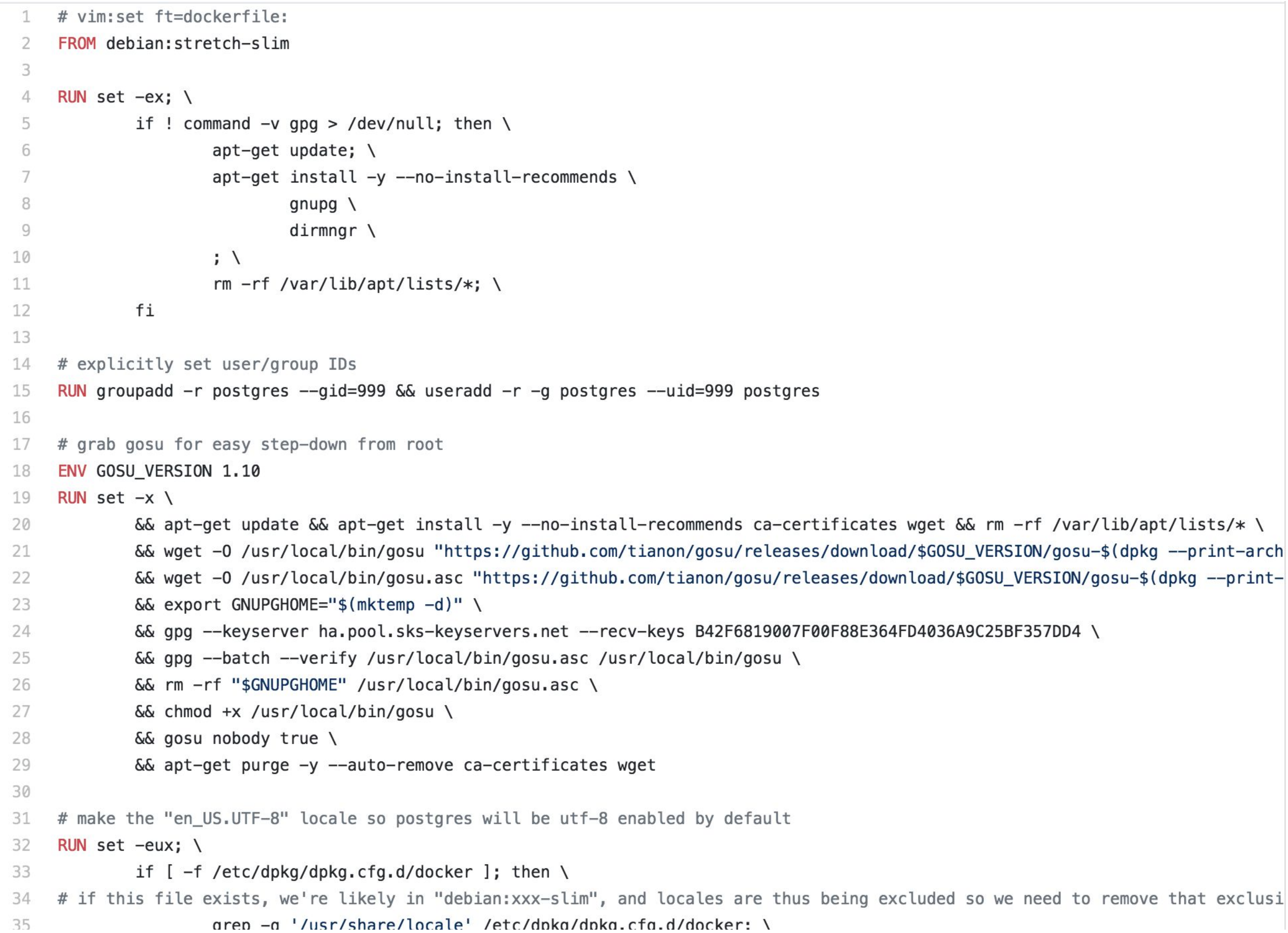
Run

Build/Run Implicit Dependency



Dockerfile-less

- An image is worth 1000 words



```
# vim:set ft=dockerfile:
FROM debian:stretch-slim

RUN set -ex; \
        if ! command -v gpg > /dev/null; then \
                apt-get update; \
                apt-get install -y --no-install-recommends \
                        gnupg \
                        dirmngr \
                ; \
                rm -rf /var/lib/apt/lists/*; \
        fi

# explicitly set user/group IDs
RUN groupadd -r postgres --gid=999 && useradd -r -g postgres --uid=999 postgres

# grab gosu for easy step-down from root
ENV GOSU_VERSION 1.10
RUN set -x \
        && apt-get update && apt-get install -y --no-install-recommends ca-certificates wget && rm -rf /var/lib/apt/lists/* \
        && wget -O /usr/local/bin/gosu "https://github.com/tianon/gosu/releases/download/$GOSU_VERSION/gosu-$(dpkg --print-arch
        && wget -O /usr/local/bin/gosu.asc "https://github.com/tianon/gosu/releases/download/$GOSU_VERSION/gosu-$(dpkg --print-
        && export GNUPGHOME="$(mktemp -d)" \
        && gpg --keyserver ha.pool.sks-keyservers.net --recv-keys B42F6819007F00F88E364FD4036A9C25BF357DD4 \
        && gpg --batch --verify /usr/local/bin/gosu.asc /usr/local/bin/gosu \
        && rm -rf "$GNUPGHOME" /usr/local/bin/gosu.asc \
        && chmod +x /usr/local/bin/gosu \
        && gosu nobody true \
        && apt-get purge -y --auto-remove ca-certificates wget

# make the "en_US.UTF-8" locale so postgres will be utf-8 enabled by default
RUN set -eux; \
        if [ -f /etc/dpkg/dpkg.cfg.d/docker ]; then \
# if this file exists, we're likely in "debian:xxx-slim", and locales are thus being excluded so we need to remove that exclusi
                grep -q '/usr/share/locale' /etc/dpkg/dpkg.cfg.d/docker; \
```

Excerpt from library/postgres:10

Daemon-less

- Performance
- Better suited for CI/CD
- Step towards decoupling
- Daemon-less build tools
    - projectatomic/buildah
    - genuinetools/img
    - GoogleContainerTools/kaniko

projectatomic/buildah

- No docker daemon involved
- Can build from dockerfile
- Can build imperatively through CLI
- CLI surface looks a lot like dockerfile
  - buildah add
  - buildah copy
  - buildah from
  - buildah run
  - buildah mount
  - buildah umount
  - buildah commit

genuinetools/img

- *The commands/UX are the same as docker {build, push, pull, login}*
- *Standalone, daemon-less, unprivileged Dockerfile and OCI compatible container image builder*
- Uses runc rootless containers

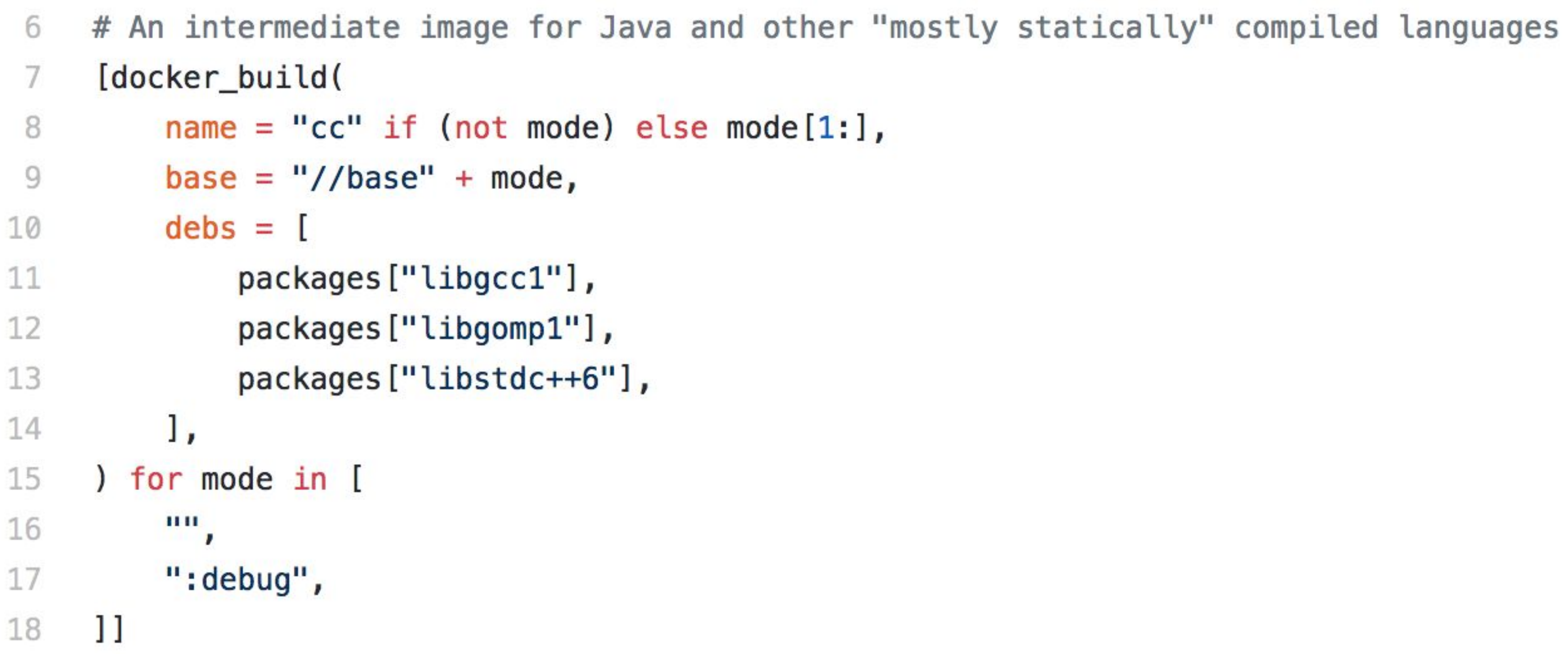
```
$ img -h
Usage: img <command>

Commands:

  build    Build an image from a Dockerfile.
  du       Show image disk usage.
  ls       List images and digests.
  login    Log in to a Docker registry.
  pull     Pull an image or a repository from a
registry.
  push     Push an image or a repository to a
registry.
```

Runtime-less

- More portable
  - No dependency on linux namespaces or cgroups
- Less complexity
- Easier to nest inside of existing containerized environments

GoogleContainerTools/distroless

- Declarative and reproducible
    - Strips timestamps
    - All dependencies known at build time
- Can’t interpret a dockerfile
- Rebase-able
- Minimal images
    - Pros:
        - Nothing but your application and runtime dependencies inside
    - Cons:
        - Nothing but your application and runtime dependencies inside



```
# An intermediate image for Java and other "mostly statically" compiled languages
[docker_build(
    name = "cc" if (not mode) else mode[1:],
    base = "//base" + mode,
    debs = [
        packages["libgcc1"],
        packages["libgomp1"],
        packages["libstdc++6"],
    ],
) for mode in [
    "",
    ":debug",
]]
```

GoogleContainerTools/kaniko

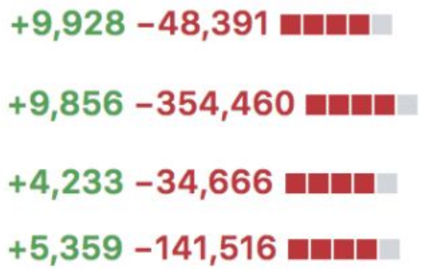
- Interprets dockerfiles
- Meant exclusively for running inside containerized environment (e.g. Kubernetes)
- Snapshots layers “naively” without union FS
  - Similar to containerd naive snapshotter or VFS
- No runtime or nested containers
- gVisor (runsc) + kaniko + kubernetes = secure on-cluster builds

Libraries

- Library for all of these tools
  - containers/image
  - google/go-containerregistry
  - google/container-registry
- Construct and manipulate images programmatically
- Can serve as the basis for alternative “frontends”

google/go-containerregistry

- Used by our libraries for image and registry actions
  - container-diff
  - skaffold
  - kubernetes/minikube
  - kaniko
  - distroless (google/containerregistry)



+9,928 −48,391 ■■■■□

+9,856 −354,460 ■■■■□

+4,233 −34,666 ■■■■□

+5,359 −141,516 ■■■■□

- Build images from scratch
- Mutate images
  - append
  - rebase
  - flatten
  - retag

What’s next?

- Reproducibility
- Decoupling build from push/pull
- Language specific optimizations
- New “frontends”
- Flexible builds without a runtime
- CRI is to Run as ??? is to Build

Dockerfile Alternatives

- Container-native package managers
  - Reuse existing package managers
  - Reproducible
- Builds without RUN are HARD
- Smart construction of layers and base images
  - Caching
  - Composability

Language specific workflows

- Optimized for frameworks and languages
- Build packs
- GoogleCloudPlatform/jib
  - Smarter layering
  - Java workflow
  - By default use distroless images
- “Faster than Light Builds”
  - Treats language level packages as layers
  - GoogleCloudPlatform/runtimes-common

Thank you!

mrick@google.com

@mattrickard on twitter