ENGEL&VÖLKERS
TECHNOLOGY

Christian Roggia, Engel & Völkers Technology GmbH

Reproducible development and deployment with Bazel and Telepresence

ENGEL&VÖLKERS TECHNOLOGY

Who is presenting today?

Christian Roggia

Software Engineer at Engel & Völkers Technology GmbH

christian.roggia@engelvoelkers.com

christian-roggia

@ChristianRoggia

Maintainer of “**Logstash**” and “**Elastic-Stack**” official Helm charts.

Owner of the “**Drone CI**” official Helm chart.

ENGEL&VÖLKERS TECHNOLOGY

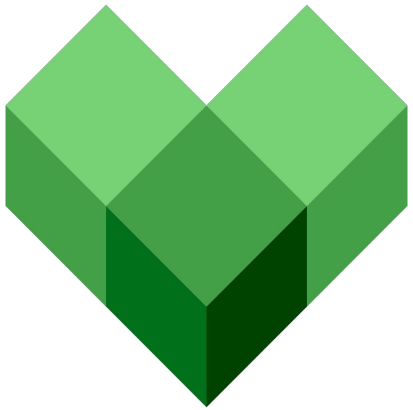
What we will discuss today?

1. **Reproducible builds with Bazel**
   Building always the same binaries everywhere.

2. **Interactive development with Telepresence**
   Bringing the Cloud on your machine.

3. **Containerized development environment**
   Running the same code everywhere.

4. **Run local containerized applications in the Cloud**
   Reproducible deployments from your machine to the Cloud.



Bazel

Version 0.25 Beta

https://bazel.build/

Build Automation Tool

Telepresence

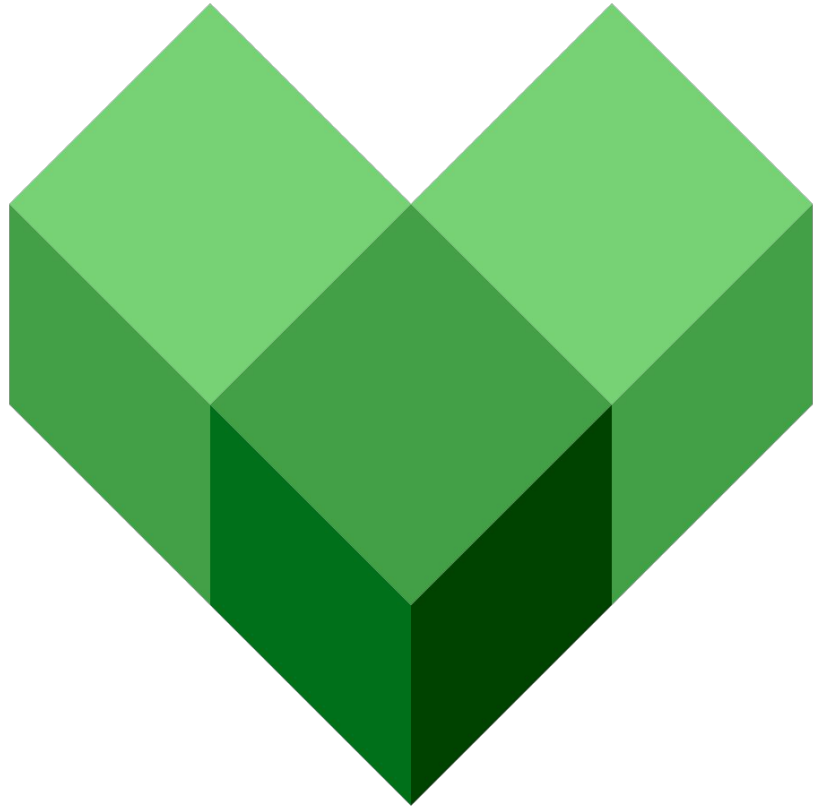
Version 0.99 Beta

https://www.telepresence.io/

Kubernetes Two-Way Proxy

Building the same binaries everywhere.

Reproducible builds with Bazel



ENGEL&VÖLKERS TECHNOLOGY

Understanding the different compilation patterns

Standard architecture

- ≈ Compilation is executed via scripts and Makefiles
- ✗ Tools and deps must be pre-installed and pre-configured in the system
- ✗ Builds are hard to reproduce
- ✗ Builds are not hermetic
- ✗ Implementation is different for each development environment

Containerized architecture

- ≈ Compilation is executed inside Docker via scripts and Makefiles
- ≈ Tools and deps must be manually downloaded and configured in the image
- ≈ Builds are not always reproducible
- ≈ Builds are not fully hermetic
- ✓ Implementation is the same for every development environment

Reproducible architecture

- ✓ Compilation is executed via Bazel, Bazel itself can be dockerized
- ✓ Tools and deps are automatically downloaded and configured by Bazel
- ✓ All builds are reproducible
- ✓ All builds are hermetic
- ✓ Implementation is the same for every development environment

Reproducible builds, also known as deterministic compilation, always produce the same binaries.

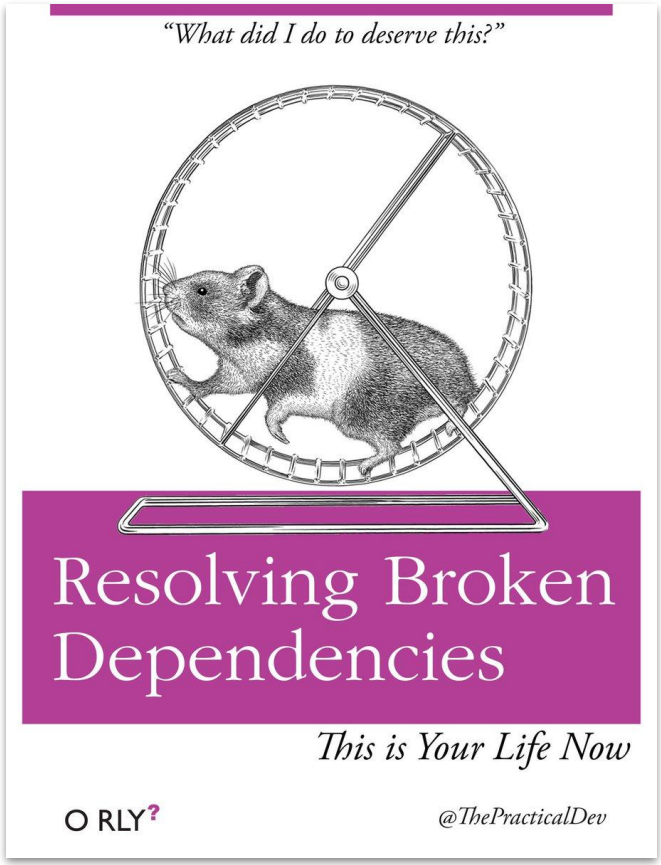
Hermetic builds are insensitive to the libraries and other software installed on the build machine or image.

Building the same binaries everywhere.

Reproducible builds with Bazel DEMO

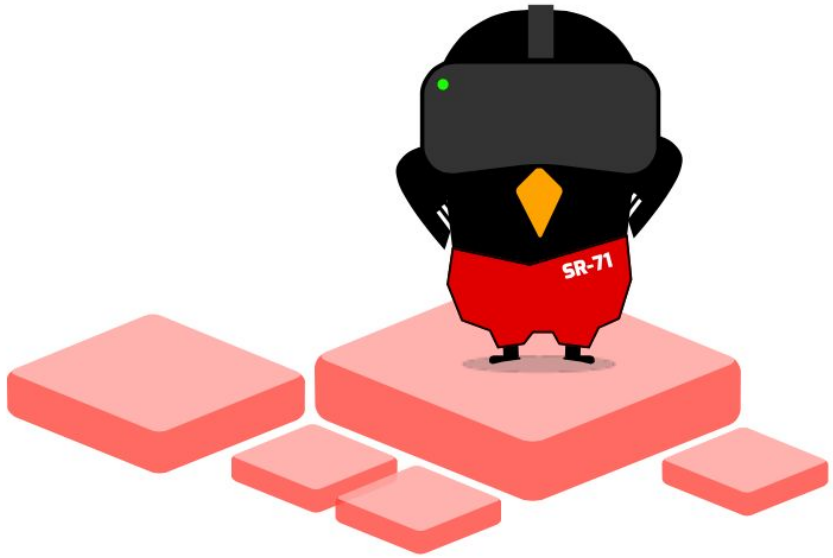
```
bazel run :gazelle
bazel run :gazelle -- update-repos -from_file=go.mod

bazel build //cmd/...
```



Bringing the Cloud on your machine.

Interactive development with Telepresence



ENGEL&VÖLKERS TECHNOLOGY

Telepresence will bring the Cloud in your machine

Internal Resources

Internal DNS and TCP traffic will be proxied to your machine.

Volumes

All mounted volumes will be proxied to your machine.

Environment Variables

All environment variables will be available in your machine.

ENGEL&VÖLKERS TECHNOLOGY

Understanding the different proxy methods

Forwarding traffic via VPN (default)

- ≈ A VPN will forward the traffic from your machine
- ✓ Strongly suggested for Go apps
- ✗ All processes are affected
- ✗ It doesn't interact well with other VPNs

Forwarding traffic via TCP injection

- ≈ Will inject a library in your process
- ✗ It doesn't work with applications statically compiled or that make use of custom DNS resolution
- ✓ Only affects a single process
- ✓ There is no conflict with other VPNs

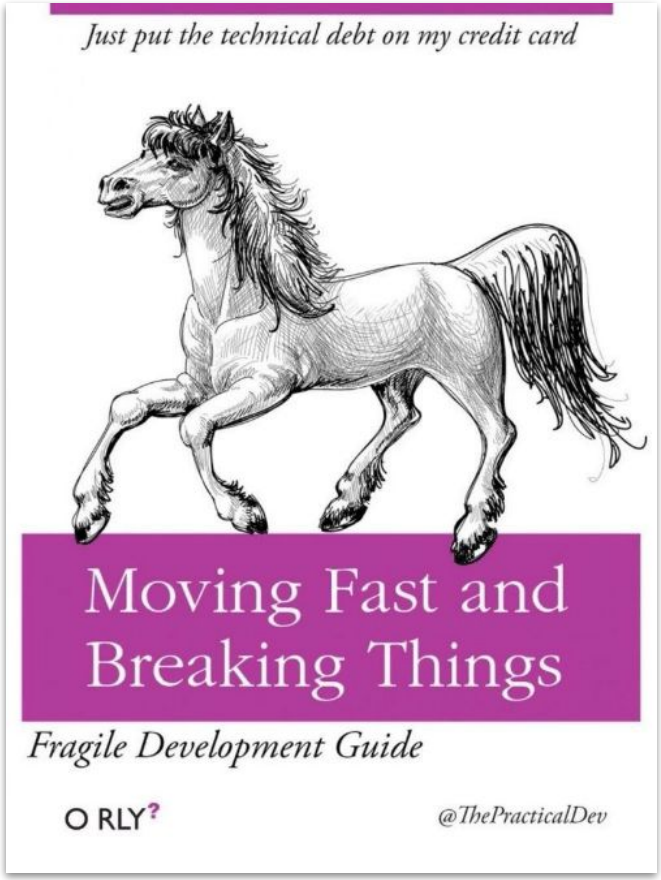
Forwarding traffic from a container

- ✓ Will proxy the traffic from the Docker container
- ✓ Allows dockerized runtime environments
- ✓ Only affects the container running the application
- ✓ There is no conflict with other VPNs

Bringing the Cloud on your machine.

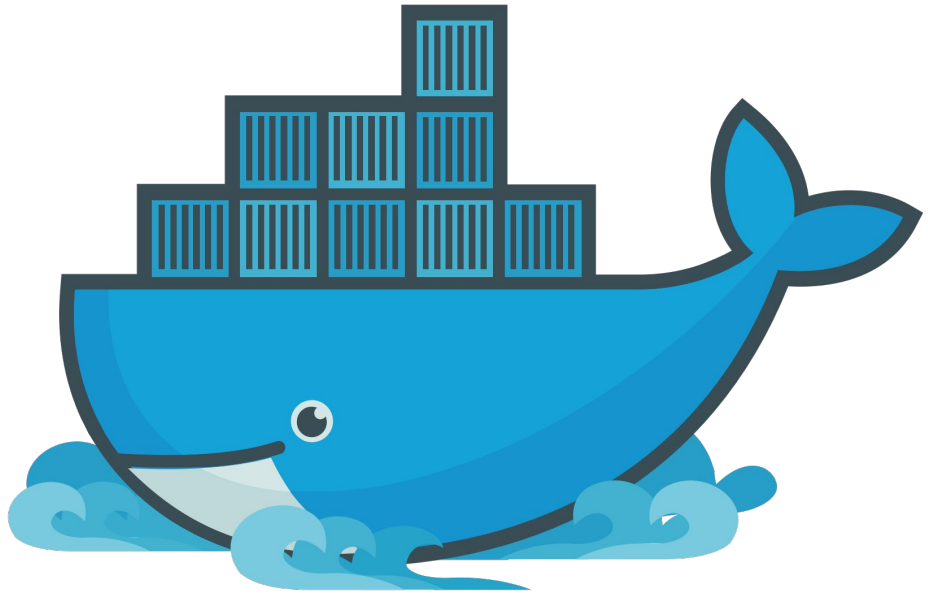
Interactive development with Telepresence DEMO

```
telepresence
telepresence --swap-deployment demo-app
telepresence --docker-run -it debian
```



Running the same code everywhere.

Containerized development environment



ENGEL&VÖLKERS TECHNOLOGY

Understanding the different containerized environments

Non-containerized environment

- ✗ Source code is built from tools configured and installed on the local machine
- ✗ Compilation will depend on the OS and on the host environment
- ✗ No isolation during compilation
- ✗ No isolation during execution
- ✗ Multiple services must run at the same time on the local machine for a full local environment

Containerized environment

- ≈ Source code is built from tools configured and installed inside the container
- ≈ Compilation will depend only on build arguments
- ✓ Compilation is isolated
- ✓ Execution is isolated
- ≈ Multiple services can run in isolated environments and interaction is orchestrated by docker-compose

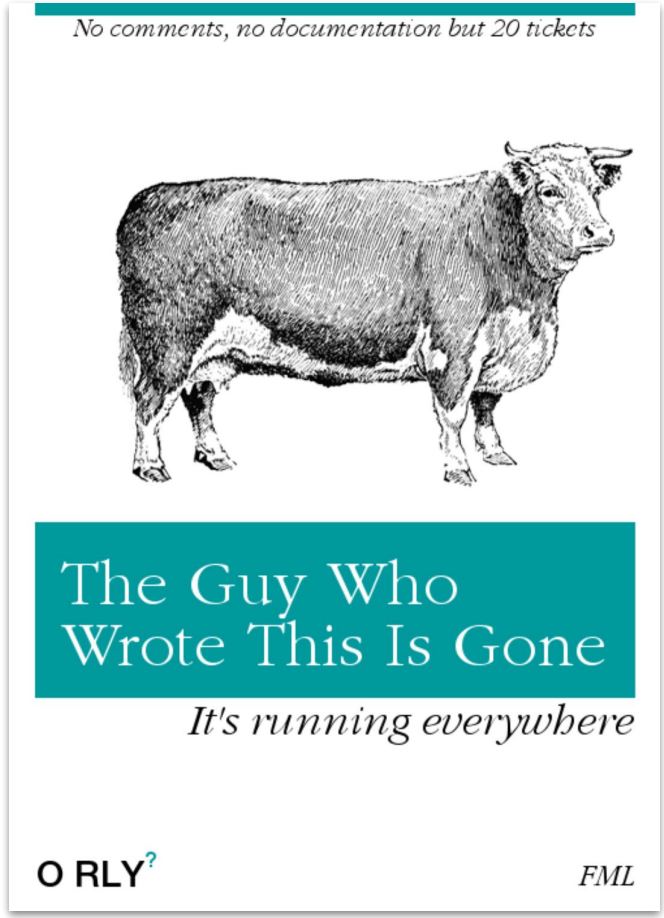
Reproducible environment

- ✓ Source code is built from Bazel inside a container
- ✓ Compilation will depend only on Bazel configuration
- ✓ Compilation is fully isolated, fully hermetic and fully reproducible
- ✓ Execution is isolated
- ≈ Multiple services can run in isolated environments and interaction is orchestrated by docker-compose

Running the same code everywhere.

Containerized development environment DEMO

```
docker system prune --all
docker build -t kubecon-bazel -f build/Dockerfile .
docker run kubecon-bazel
```



Reproducible deployments from your machine to the Cloud.

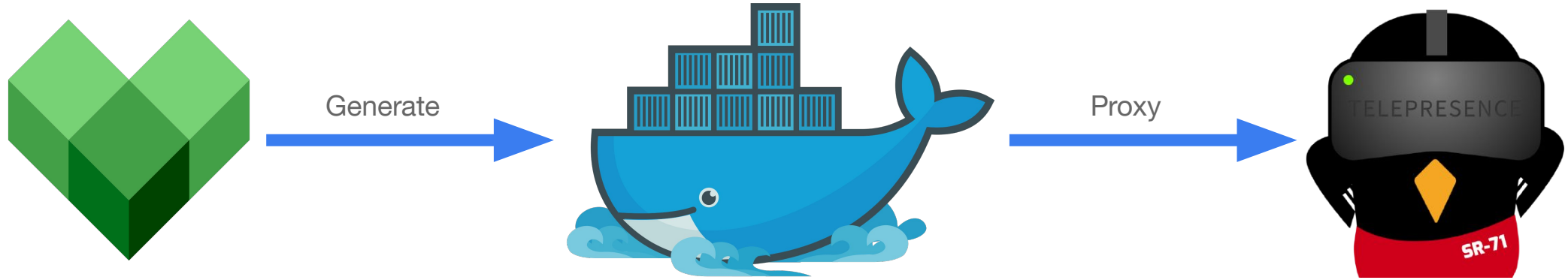
Run local containerized applications in the Cloud

ENGEL&VÖLKERS TECHNOLOGY

Feeding reproducible binaries to Telepresence

Both Bazel and Telepresence can be combined in a containerized environment:

- A special kind of images, called distroless, are generated by Bazel.
- Docker images generated by Bazel are designed to be reproducible.
- These images can be fed to Telepresence via container traffic forwarding.



Reproducible Builds | Distroless Images | Reproducible Deployments

ENGEL&VÖLKERS TECHNOLOGY

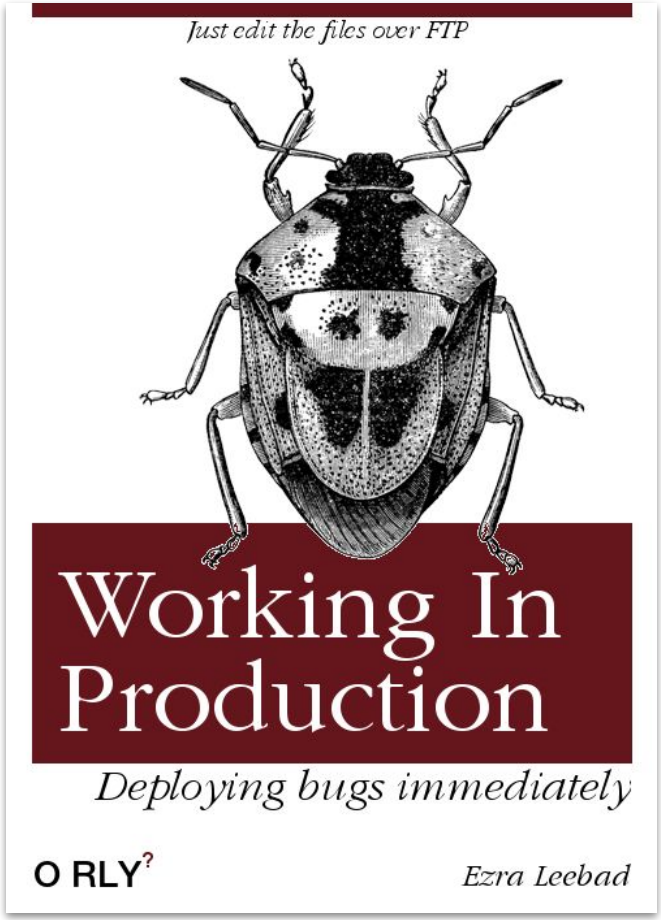
Fully reproducible development and deployment

- ➔ Compilation and execution are isolated.
- ➔ Builds are hermetic and reproducible.
- ➔ Projects will build and run out-of-the-box.
- ➔ Resources, volumes and environment variables are proxied to your machine.
- ➔ Tools like docker-compose can be omitted for large and complex applications.
- ➔ Makefiles and bash scripts can be replaced by declarative configurations.
- ➔ Tools installation and pre-configuration is no longer required for developers.

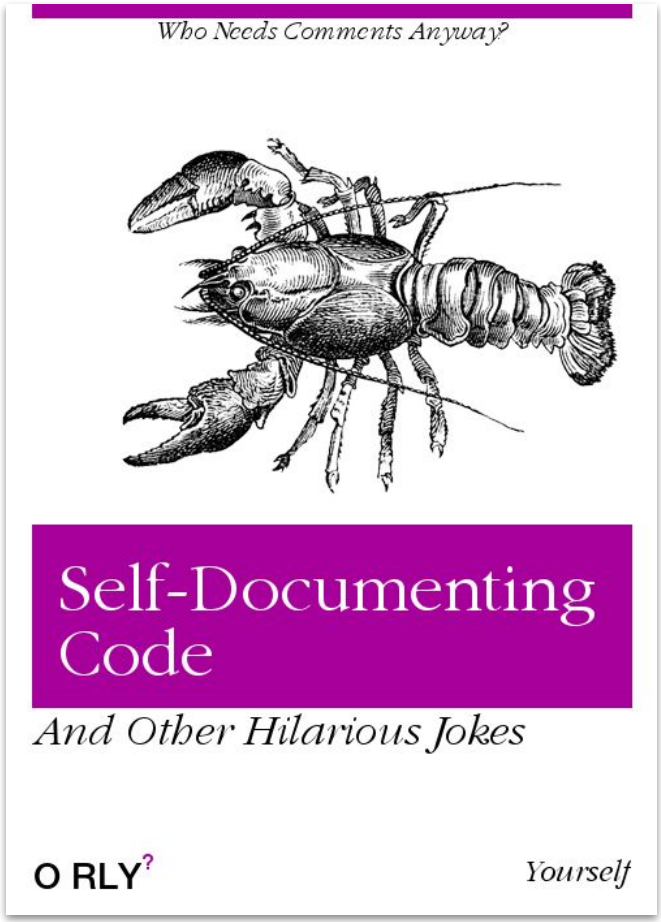
Reproducible deployments from your machine to the Cloud.

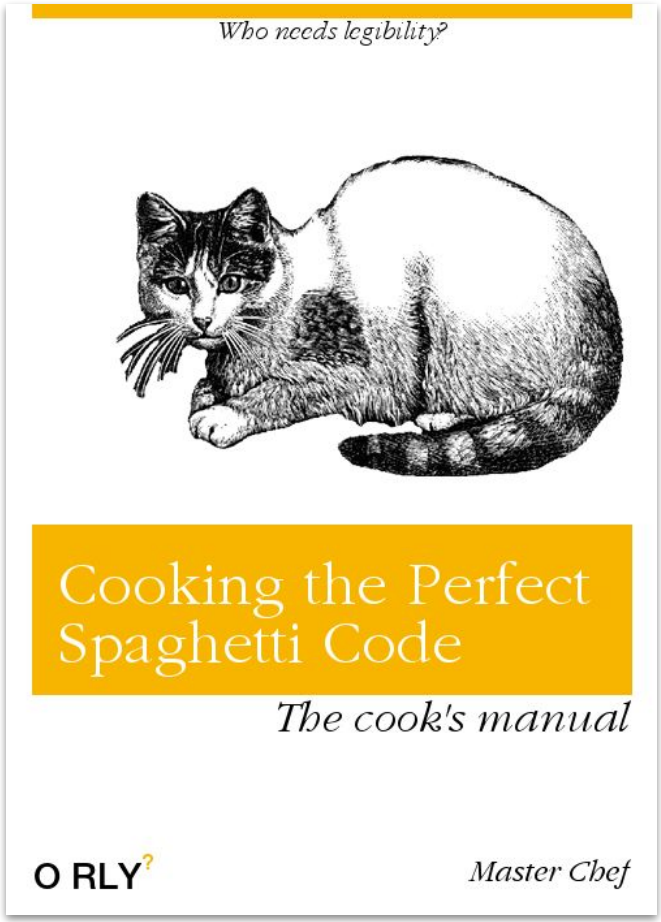
Run local containerized applications in the Cloud DEMO

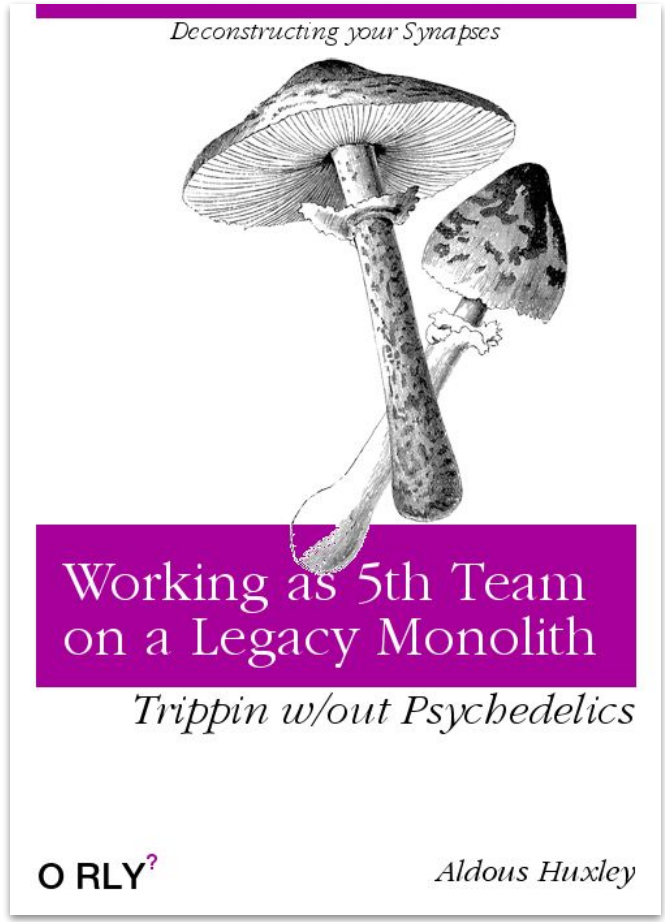
```
bazel run :register-image
telepresence --docker-run k8s.io/kubecon-bazel-telepresence:latest
```



There will be virtually no difference between your local deployment and a remote deployment


Who Needs Comments Anyway?
Self-Documenting Code
And Other Hilarious Jokes
O RLY?
Yourself


Who needs legibility?
Cooking the Perfect Spaghetti Code
The cook's manual
O RLY?
Master Chef


Deconstructing your Synapses
Working as 5th Team on a Legacy Monolith
Trippin w/out Psychedelics
O RLY?
Aldous Huxley

Thank you for your attention!