Please

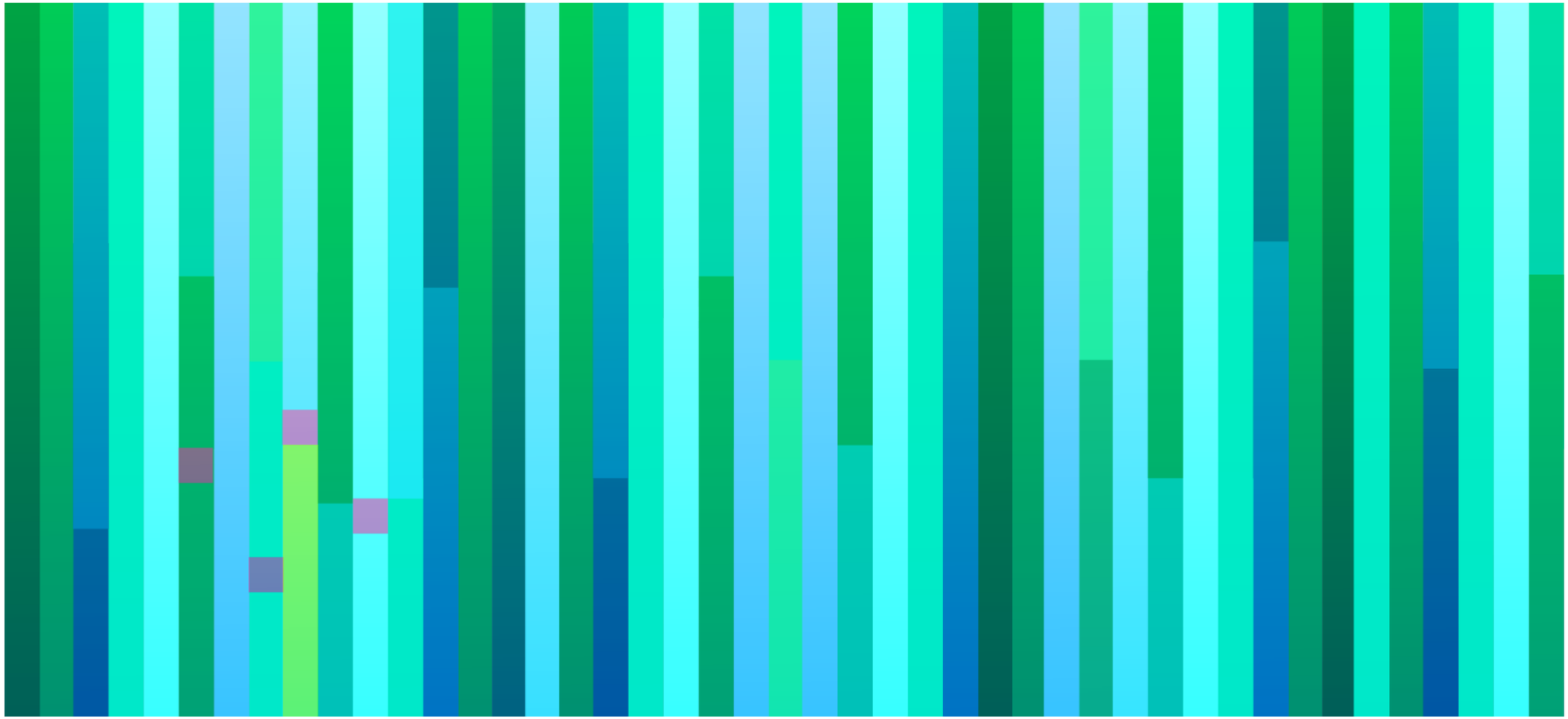
Ask questions through the app

Rate Session

Thank you!

@GOTOber

gotober.com



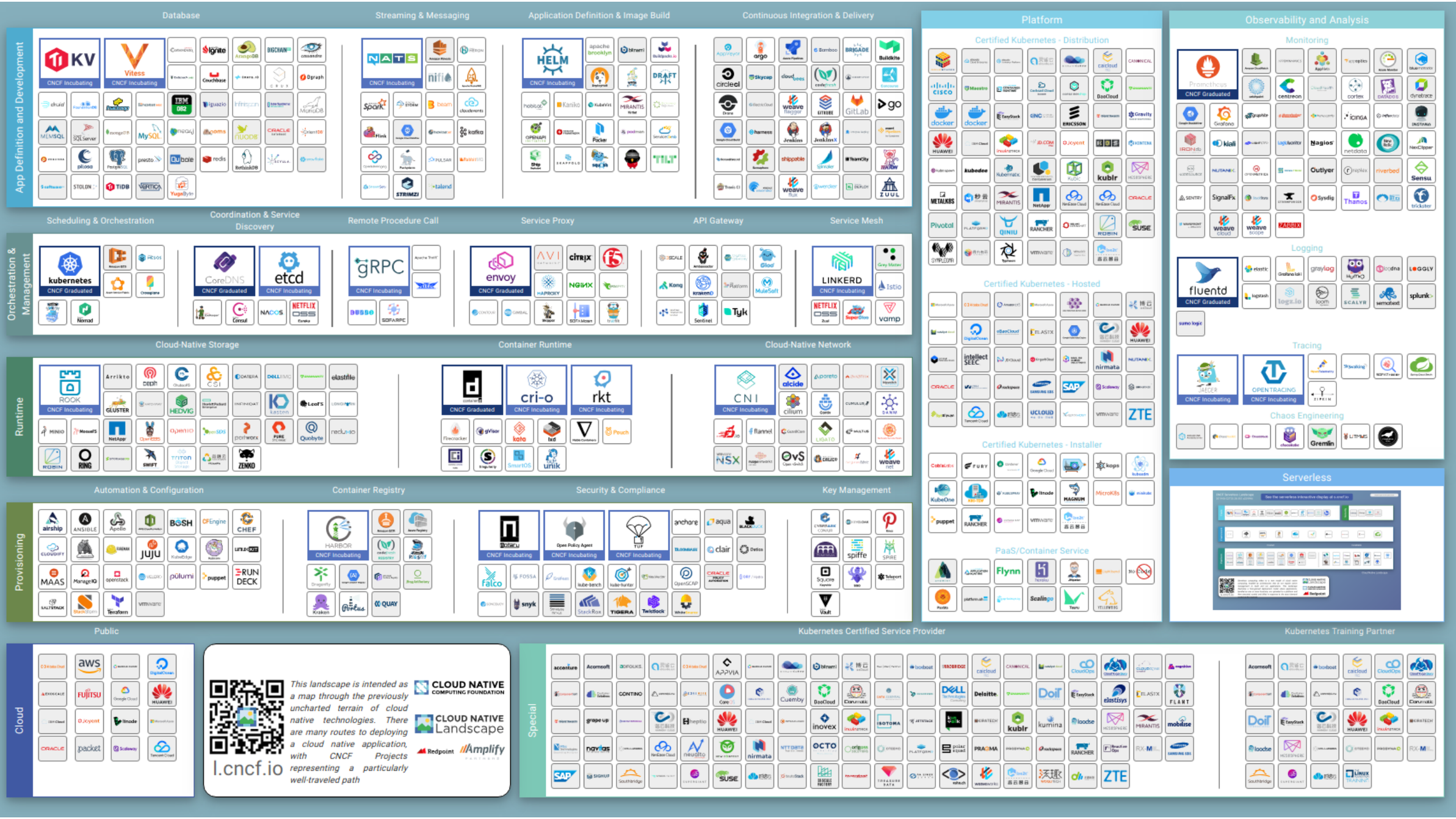
The State of Kubernetes Development Tooling

October 2019

garden

> *Kubernetes the hard way* falsely implies that there's an easy way.

@pczarkowski


App Definition and Development
Database
Streaming & Messaging
Application Definition & Image Build
Continuous Integration & Delivery
Orchestration & Management
Scheduling & Orchestration
Coordination & Service Discovery
Remote Procedure Call
Service Proxy
API Gateway
Service Mesh
Runtime
Cloud-Native Storage
Container Runtime
Cloud-Native Network
Provisioning
Automation & Configuration
Container Registry
Security & Compliance
Key Management
Cloud
Public
Platform
Certified Kubernetes - Distribution
Certified Kubernetes - Hosted
Certified Kubernetes - Installer
PaaS/Container Service
Observability and Analysis
Monitoring
Logging
Tracing
Chaos Engineering
Serverless
Special
Kubernetes Certified Service Provider
Kubernetes Training Partner
CNCF Incubating
CNCF Graduated
l.cncf.io
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path
CLOUD NATIVE COMPUTING FOUNDATION
CLOUD NATIVE Landscape
Redpoint
Amplify

Who is this talk for?

Developers working on apps that...

- Run **on** Kubernetes.
- Are **not** Kubernetes itself. E.g. CRDs & operators.

Who is this talk for?

Developers looking to improve...

- Their **pre-commit** workflow.
- The part of their day where **code** gets written.

But not necessarily...

- Their CI/CD pipelines.
- Ops stuff not directly related to writing code.

> You put the whole app in one container?!

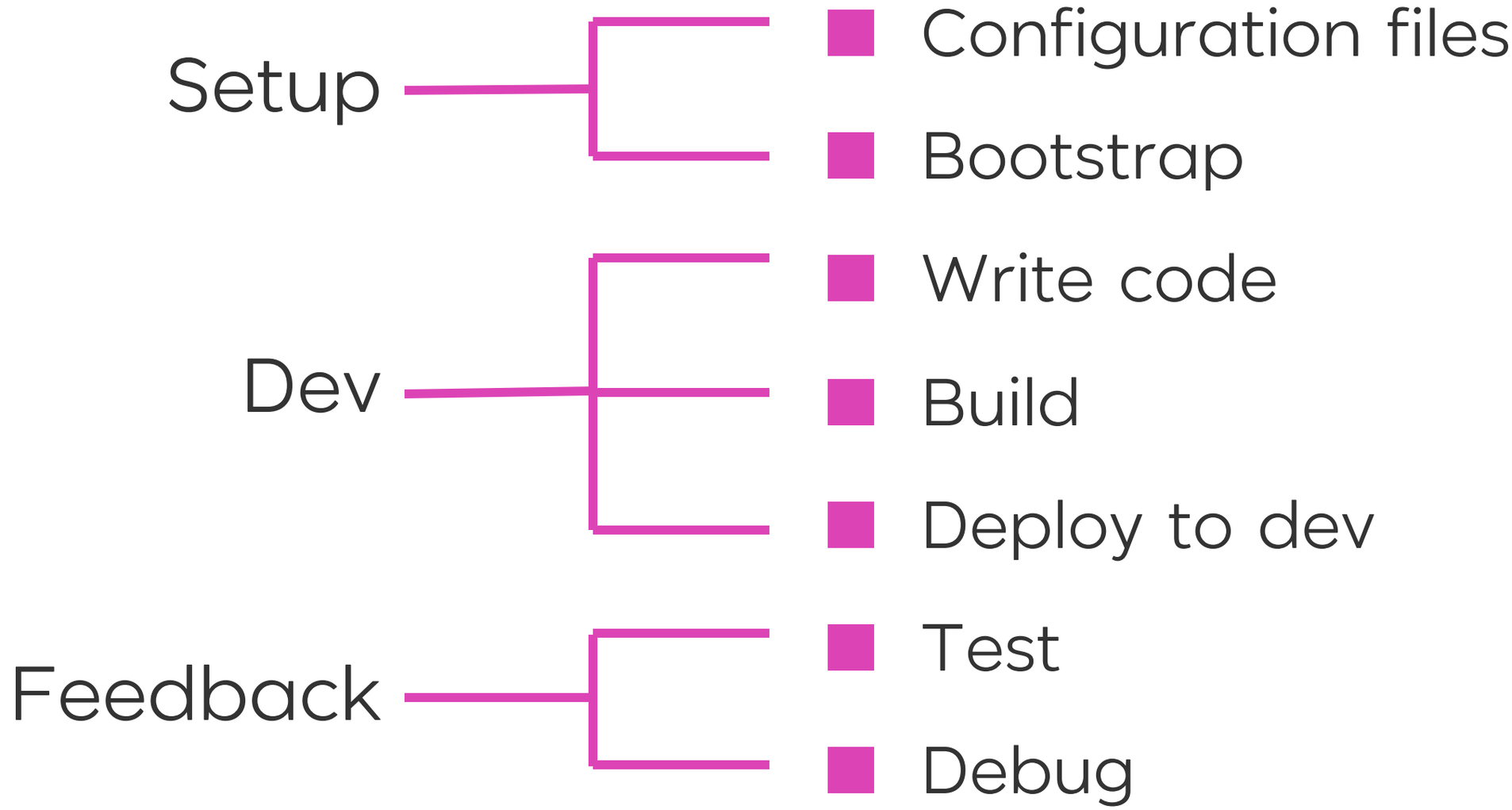
@7hunderbird

Development Cycle:

- Provisioning
- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug
- Git & CI
- Ship to prod. & CD
- Observability

Pre-Commit Stage:

- ~~Provisioning~~
- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug
- ~~Git & CI~~
- ~~Ship to prod. & CD~~
- ~~Observability~~


Setup
Configuration files
Bootstrap
Dev
Write code
Build
Deploy to dev
Feedback
Test
Debug

garden

Wishlist

- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug

- Manageable size
- Create them for me?

Wishlist

- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug

- Setup tasks (like migrations)

Wishlist

- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug

- Automate docker/kubectl cycle
- Short feedback loop
- Unobtrusive
- Hot reload
- Easy to on-board
- Dependency-aware

Wishlist

- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug

- Do testing
- Highlight failures
- End-to-end tests

garden

Wishlist

- Configuration files
- Bootstrap
- Write code
- Build
- Deploy to dev
- Test
- Debug

- Log streaming
- Highlight issues
- Allow breakpoints
- Not tracing

> “Maybe containers were a mistake.”

@jessfraz

Categories

- Debug
- Connect
- Build & Deploy Cycle

Categories

- Debug
- Connect
- Build & Deploy Cycle

Tools

- Draft
- Garden
- Skaffold
- Tilt
- Jenkins X
- Forge
- Helm
- ksync
- kubefwd
- Okteto
- Telepresence
- inlets
- Squash
- Stern

Others...

- kubeval
- yq
- kubectx
- kubens
- ytt
- kpxg
- kbld
- kapp
- kwt
- ...

Others...

Trivia:
Which of these
isn't a real tool?

- kubeval
- yq
- kubectx
- kubens
- ytt
- kpxg
- kbld
- kapp
- kwt
- ...

“Kubernetes: complicating simple things since 2014.”

@texmandie

Debug

- **kubectl**
- Squash
- Stern

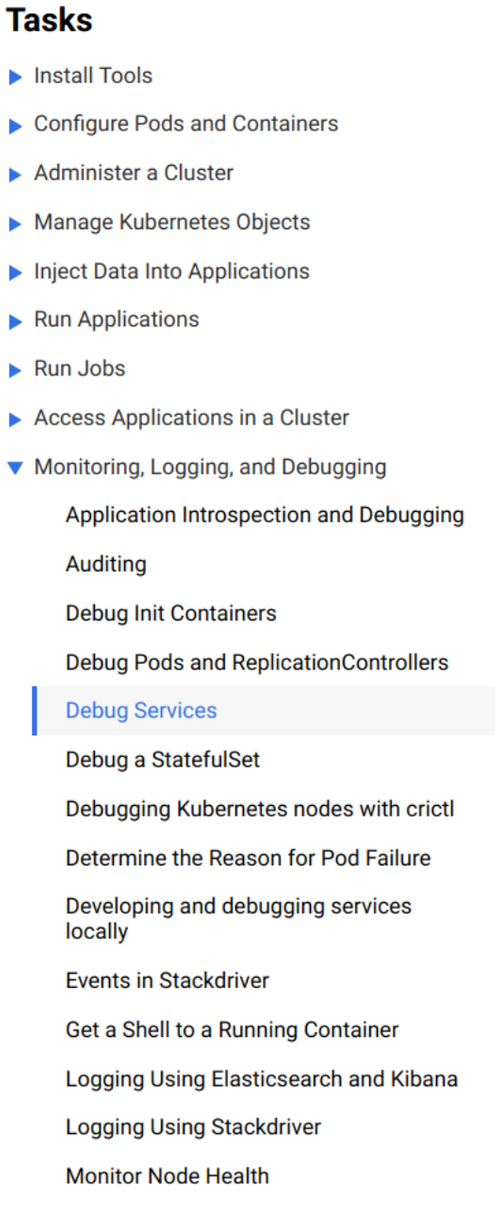
Notes

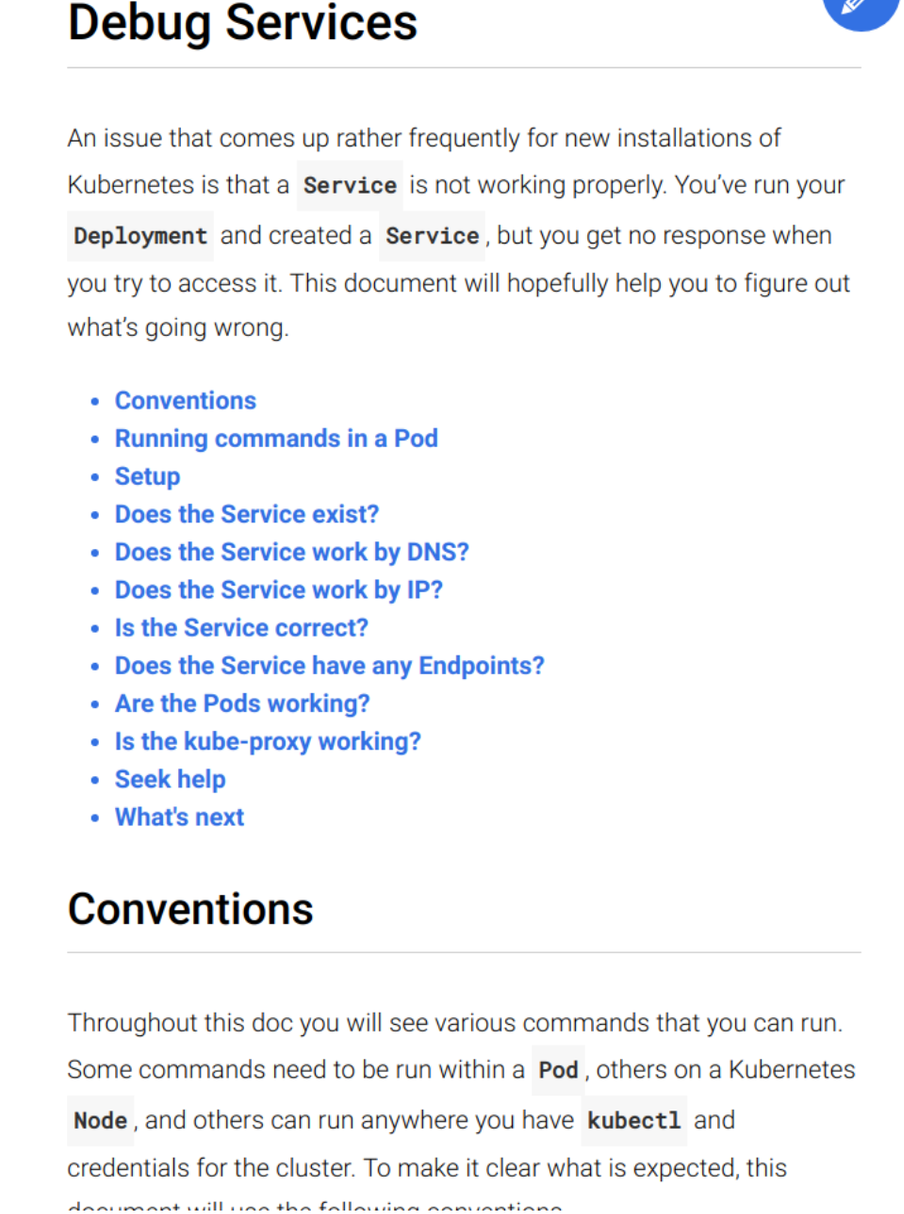
- Easily shows you the status of your pods, deployments, services.
- Learn it.
- **kubectl logs --follow -n [namespace] [pod]**

garden

Debug

- kubectl
- Squash
- Stern





Debug Services

An issue that comes up rather frequently for new installations of Kubernetes is that a `Service` is not working properly. You've run your `Deployment` and created a `Service`, but you get no response when you try to access it. This document will hopefully help you to figure out what's going wrong.

- Conventions
- Running commands in a Pod
- Setup
- Does the Service exist?
- Does the Service work by DNS?
- Does the Service work by IP?
- Is the Service correct?
- Does the Service have any Endpoints?
- Are the Pods working?
- Is the kube-proxy working?
- Seek help
- What's next

Conventions

Throughout this doc you will see various commands that you can run. Some commands need to be run within a `Pod`, others on a Kubernetes `Node`, and others can run anywhere you have `kubectl` and credentials for the cluster. To make it clear what is expected, this

Debug

- kubectl
- **Squash**
- Stern

Notes:

- Steps and breakpoints on deployed containers!
- No more Printf debugging 🙌
- IDE or CLI
- Delve, gdb, Java, Node, Python
- Fun!

garden

Debug

- kubectl
- **Squash**
- Stern

garden

@ellenkorbes

```
→  nn (ndnd→master) ✗ squashctl
Attaching debugger
? Select a debugger  [Use arrows to move, type to filter]
❯ dlv
  java
  java-port
  gdb
? Select a debugger dlv
? Select a namespace to debug  [Use arrows to move, type to filter]
  default
  garden-system
  garden-system--metadata
❯ ndnd
  ndnd--metadata
  squash-debugger
? Select a namespace to debug ndnd
? Select a pod  [Use arrows to move, type to filter]
  console-6496757557-wd7b9
  frontend-77b7d68bb4-znd5d
❯ nn-66bf544fd6-fz5jl
  normalize-d5855b6f4-7g47t
  tiller-deploy-6ffb5bfc5f-6rl6k
? Select a pod nn-66bf544fd6-fz5jl
? Going to attach dlv to pod nn-66bf544fd6-fz5jl. continue? Yes
Squash will create a debugger pod in your target pod's namespace.
Creating service account squash-plank in namespace squash-debugger

Creating cluster role squash-plank-cr

Creating cluster role binding squash-plank-crb

All squashctl permission resources created.

Pod creating
Forwarding from 127.0.0.1:37909 -> 42673
Forwarding from [::1]:37909 -> 42673
Handling connection for 37909
                              Type 'help' for list of commands.
(dlv) b nn
Breakpoint 1 set at 0x73c05b for main.nn() /go/src/app/main.go:462
```

Debug

- kubectl
- Squash
- **Stern**

Notes:

- Streams logs from multiple pods
- Color-coded
- Easy to filter
- It's the best
- I love it 💜
- Use it!

Debug

- kubectl
- Squash
- **Stern**

@ellenkorbes

```
console-6496757557-cwzwn console 2019-06-19T15:00:31.529946667Z 2019/06/19 15:00:31 1
72.17.0.6:32794 304 GET /
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:31.537552186Z Connected!
console-6496757557-cwzwn console 2019-06-19T15:00:31.569155761Z 2019/06/19 15:00:31 1
72.17.0.6:32794 200 GET /auth_token.js
console-6496757557-cwzwn console 2019-06-19T15:00:31.572633237Z 2019/06/19 15:00:31 1
72.17.0.6:32804 304 GET /js/gotty.js
console-6496757557-cwzwn console 2019-06-19T15:00:31.641343394Z 2019/06/19 15:00:31 1
72.17.0.6:32806 304 GET /js/hterm.js
console-6496757557-cwzwn console 2019-06-19T15:00:31.663861678Z 2019/06/19 15:00:31 N
ew client connected: 172.17.0.6:32794
console-6496757557-cwzwn console 2019-06-19T15:00:31.704448626Z 2019/06/19 15:00:31 C
ommand is running for client 172.17.0.6:32794 with PID 42 (args=""), connections: 1
console-6496757557-cwzwn console 2019-06-19T15:00:31.70452047Z 2019/06/19 15:00:31 17
2.17.0.6:32794 101 GET /ws
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:32.746613773Z 0 cost: 0.033669, 0.500000
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:33.860414448Z 13 cost: 0.006140, 0.493539
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:34.777334223Z 26 cost: 0.006140, 0.487161
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:35.849927767Z 39 cost: 0.006140, 0.480866
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:36.77144575Z 52 cost: 0.006140, 0.474652
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:37.748666509Z 65 cost: 0.006140, 0.468518
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:38.750295572Z 78 cost: 0.006140, 0.462464
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:39.737346773Z 91 cost: 0.006140, 0.456488
nn-5bb4c94c66-pcrws nn 2019-06-19T15:00:40.750518139Z 104 cost: 0.006140, 0.450589
normalize-d5855b6f4-44skr normalize 2019-06-19T15:00:41.537877385Z 172.17.0.6:35894 n
ormal sent: ping
normalize-d5855b6f4-44skr normalize 2019-06-19T15:00:41.548921549Z 172.17.0.6:35894 n
ormal sent: ping
```

garden

Connect

- **Forge**
- Helm
- inlets
- ksync
- kubefwd
- Okteto
- Telepresence

Notes:

- "[...] think of it as a very fancy shell script that does a docker build, docker push, kubectl apply [...]" — rdldw
- Builds and deploys! Doesn't watch.
- Requires YAML, Dockerfile, Forge config.
- Allows profiles e.g. dev/canary/production.
- Development seems stopped 😢

FORGE

garden

Connect

- Forge
- **Helm**
- inlets
- ksync
- kubefwd
- Okteto
- Telepresence

Notes:

- Package manager for Kubernetes application structure.
- Install and uninstall off-the-shelf apps. Think apt-get, brew, npm, pip, gem.
- App lifecycle management. Deploy a service's new version, roll back, delete.
- Omnipresent: Haven't used it? You will.
- Pro & con: Go templating on configs.

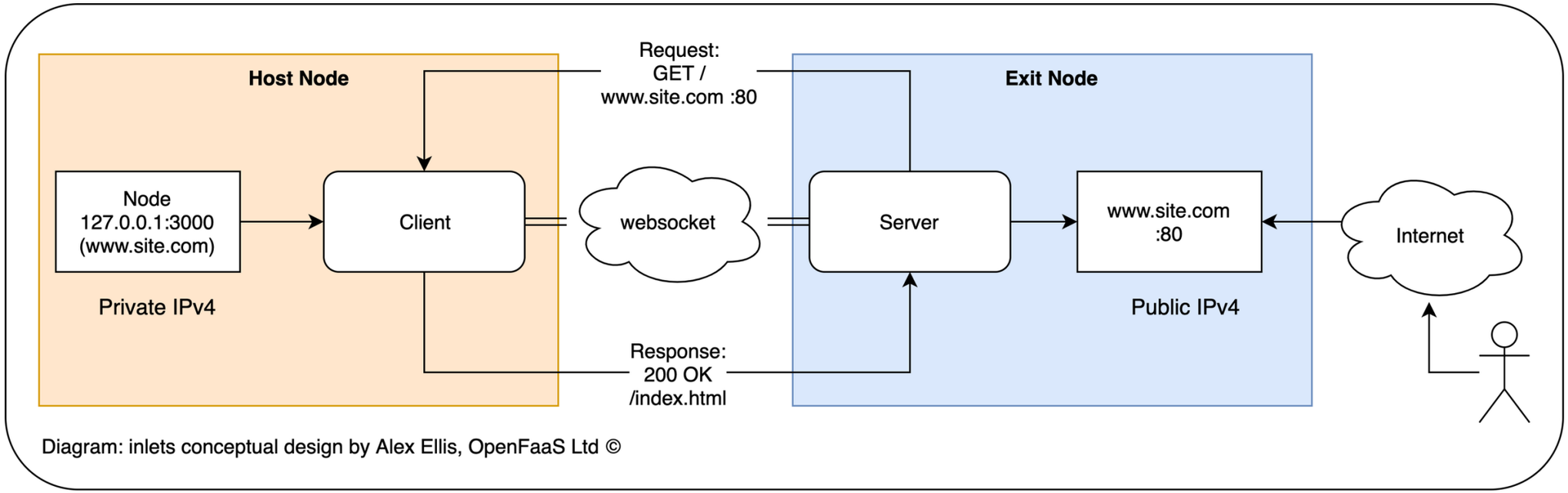
Connect

- Forge
- Helm
- **inlets**
- ksync
- kubefwd
- Okteto
- Telepresence

Notes:

- Reverse proxy & websocket tunnel to expose internal endpoints to the internet.
- Similar to ngrok or Argo Tunnel, but free and open source.
- Has a Kubernetes operator so you can seamlessly expose a local Kubernetes cluster.
- Use it to share works in progress, set up webhooks in development, and so on.
- **ACHTUNG!** You're exposing internals to the internet. Be mindful of security.

inlets



garden

Connect

- Forge
- Helm
- inlets
- **ksync**
- kubefwd
- Okteto
- Telepresence

Notes:

- Watches your file system
- Syncs local files with files in your cluster
- Straight forward and easy to use
- Built on top of Syncthing
- You do need a watcher inside the container to act on the updated files!

Connect

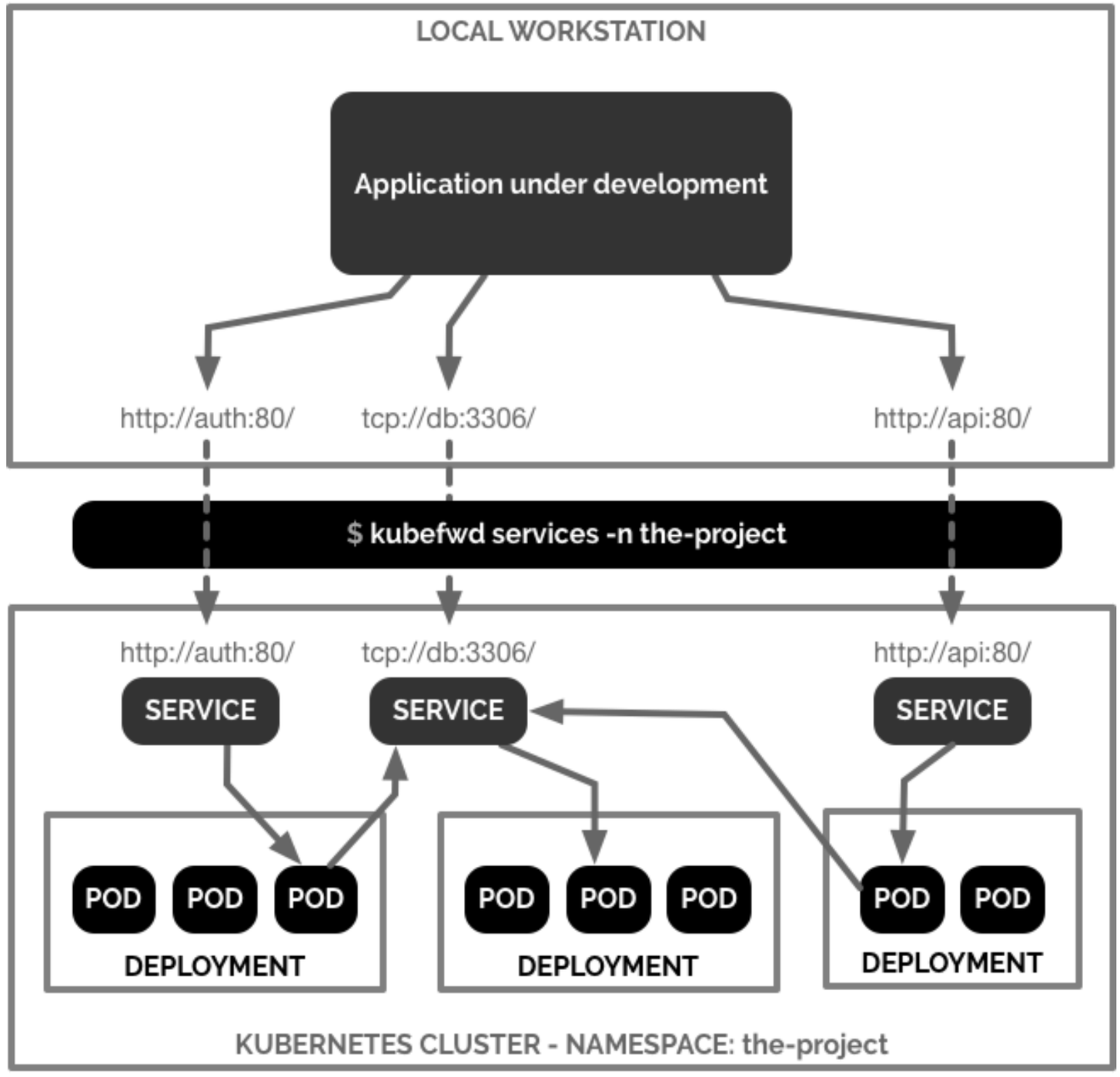
- Forge
- Helm
- inlets
- ksync
- **kubefwd**
- Okteto
- Telepresence

Notes:

- Forward all possible ports—as opposed to kubectl which forwards individual ports.
- Access any service locally like a pod in the cluster
- Easy! Fun! ~~Buy it now! (Just kidding.)~~

Connect

- Forge
- Helm
- inlets
- ksync
- **kubefwd**
- Okteto
- Telepresence



garden

Connect

- Forge
- Helm
- inlets
- ksync
- kubefwd
- **Okteto**
- Telepresence

Notes:

- Swaps a deployment for a development environment.
- Syncs & watches files, gives you a remote terminal, does port forwarding.
- Ideal to build admission controllers, operators, things deeply integrated with k8s.
- One service at a time.

Connect

- Forge
- Helm
- inlets
- ksync
- kubefwd
- Okteto
- **Telepresence**

Notes:

- Hot swaps a running pod and creates a local shell where the network is proxied into the cluster.
- Run app locally; it acts as if in the cluster.
- Use local IDE, debugger, etc.
- Vs. Squash: Single instance. Might cause conflicts like env vars and docker version.
- Vs. Okteto: Local machine pretends to be a node vs. moving dev into the cluster.

TELEPRESENCE

garden

Build & Deploy

- **Draft**
- Garden
- Jenkins X
- Skaffold
- Tilt

garden

Notes:

- Generates config files: Great if you're new to containers or Kubernetes! Helm based.
- Supports bootstrapping tasks.
- Works per-service: You must "draft up" for every service.
- "The team originally working on Draft have to put it on the backburner while focusing on efforts like Brigade, Helm 3, and CNAB."
- No hot reload!

Build & Deploy

- **Draft**
- Garden
- Jenkins X
- Skaffold
- Tilt

```
bash-3.2$ ls
glide.yaml main.go
bash-3.2$ draft create -a understanding-turkey
--> Draft detected the primary language as Go with 100.000000% certainty.
--> Ready to sail
bash-3.2$ ls
Dockerfile chart      draft.toml glide.yaml main.go
bash-3.2$ draft up
Draft Up Started: 'understanding-turkey'
understanding-turkey: Building Docker Image: SUCCESS ⚓  (11.0049s)
understanding-turkey: Pushing Docker Image: SUCCESS ⚓  (16.0054s)
understanding-turkey: Releasing Application: SUCCESS ⚓  (0.5202s)
understanding-turkey: Build ID: 01BV21PKV1MXS9P9SJYGWXV94M
bash-3.2$ draft connect &
[1] 4255
bash-3.2$ Connecting to your app...SUCCESS...Connect to your app on localhost:54997
Starting log streaming...
+ exec app

bash-3.2$ curl localhost:54997
Hello World, I'm Draft!
```

garden

Build & Deploy

- Draft
- **Garden**
- Jenkins X
- Skaffold
- Tilt

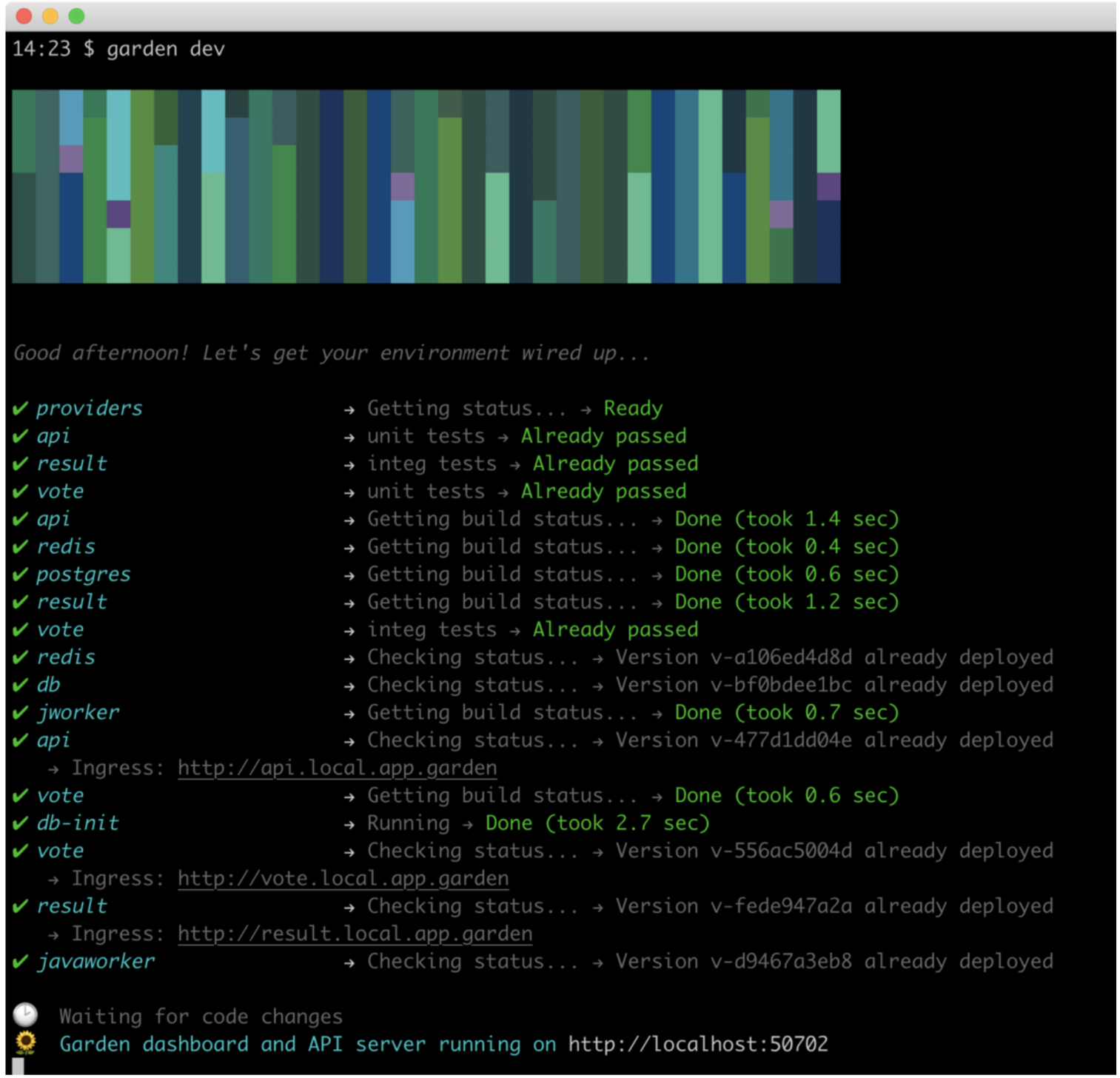
Notes:

- No Kubernetes manifests required. (Unless you want to!)
- Allows in-cluster builds. Local Kubernetes, Docker, minikube not required.
- Project awareness: It tracks build and deploy dependencies.
- Tests are first-class citizens. Locally & CI.
- Pluggable: Custom module types for e.g. OpenFaaS, Java, Terraform. (Soon also Go!)

garden

Build & Deploy

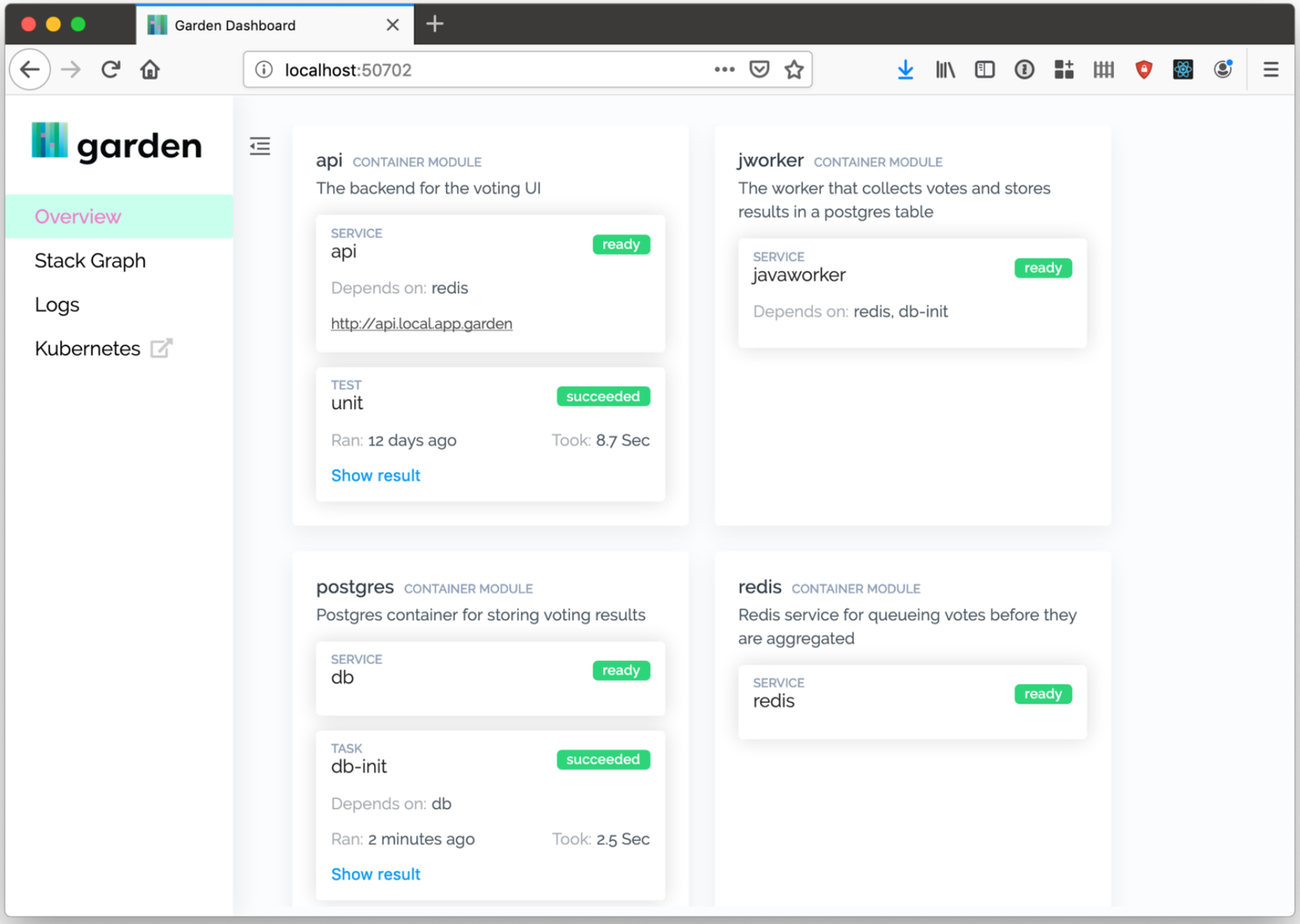
- Draft
- **Garden**
- Jenkins X
- Skaffold
- Tilt



garden

Build & Deploy

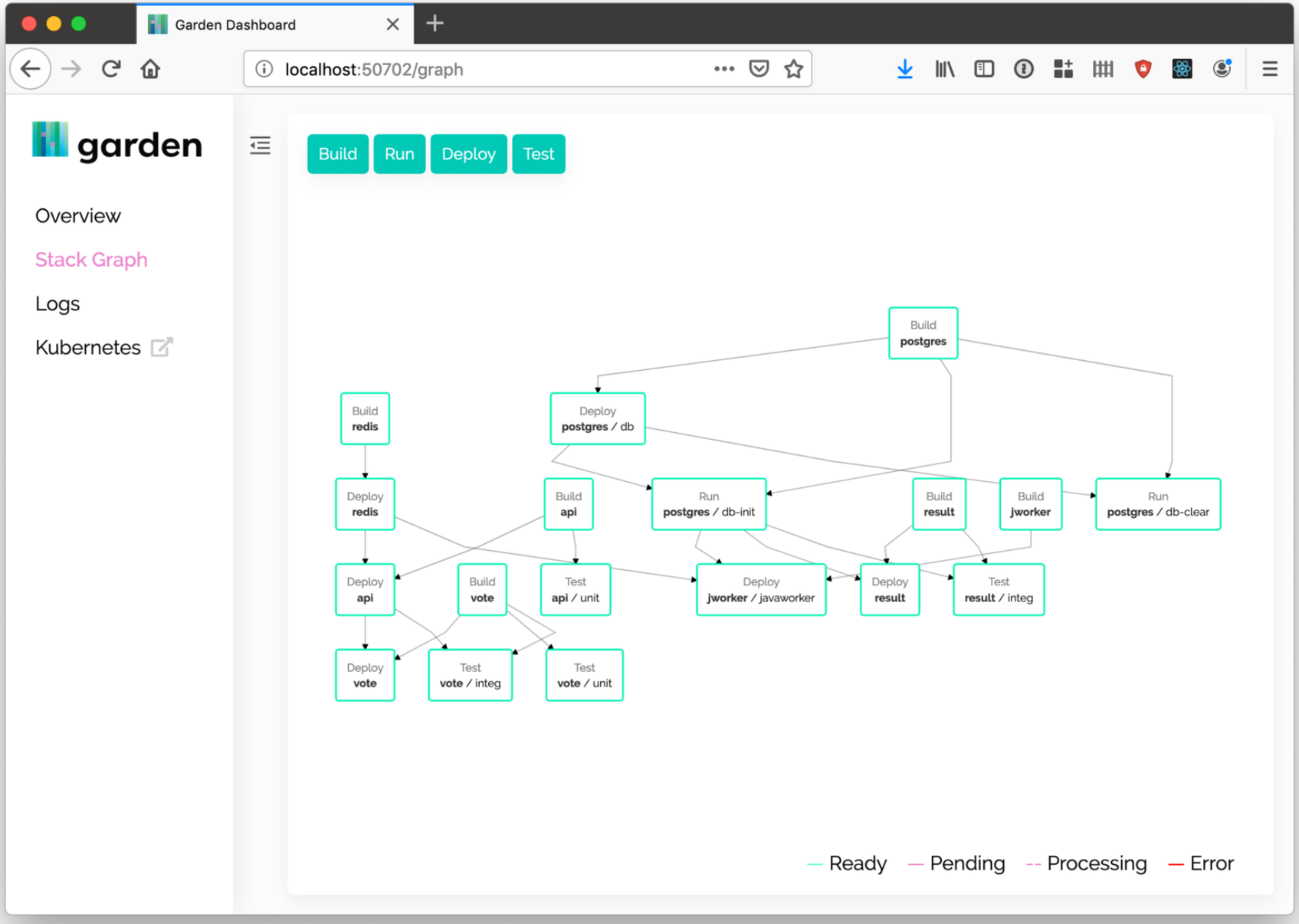
- Draft
- **Garden**
- Jenkins X
- Skaffold
- Tilt



garden

Build & Deploy

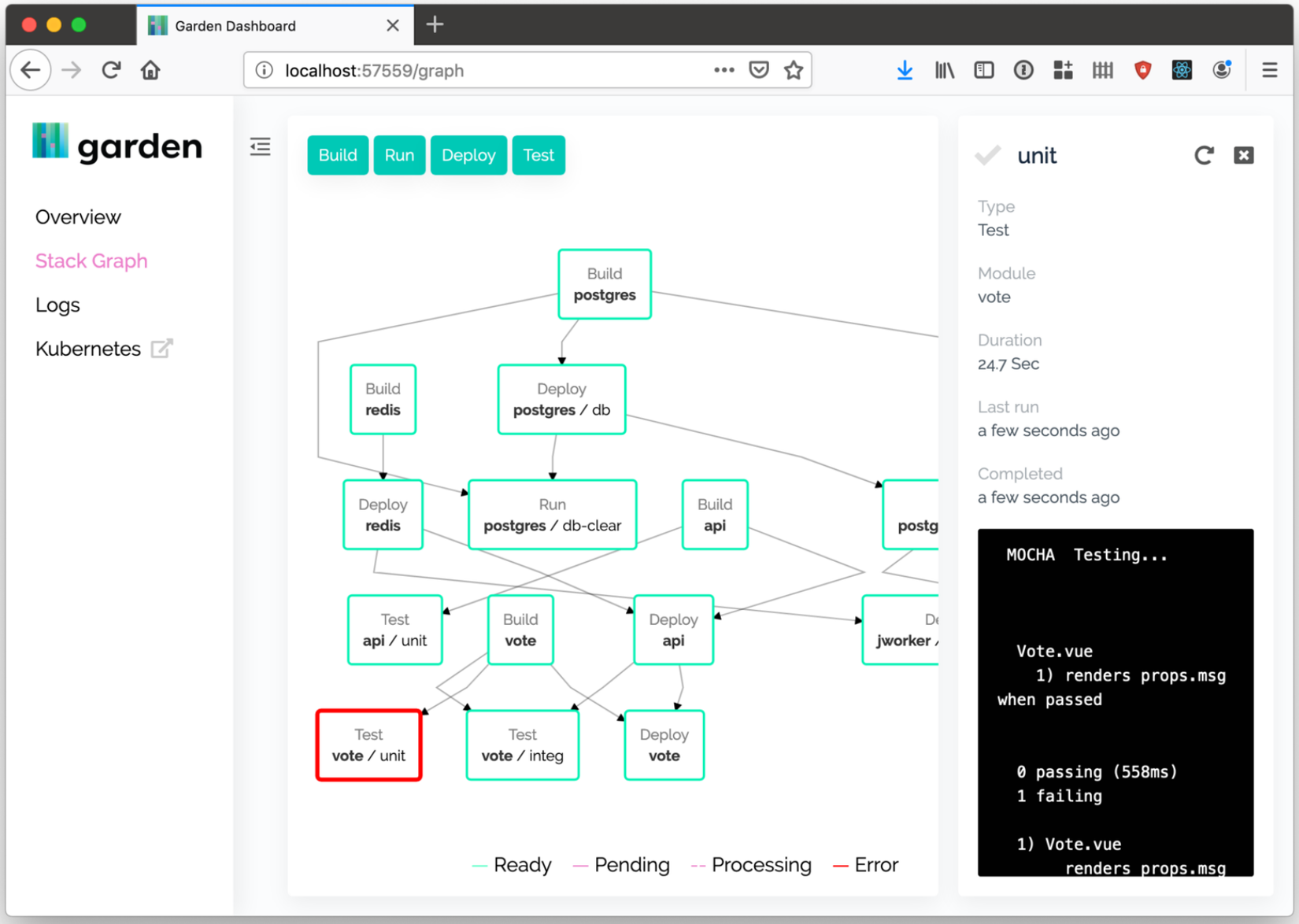
- Draft
- **Garden**
- Jenkins X
- Skaffold
- Tilt



garden

Build & Deploy

- Draft
- **Garden**
- Jenkins X
- Skaffold
- Tilt



garden

Build & Deploy

- Draft
- Garden
- **Jenkins X**
- Skaffold
- Tilt

Notes:

- Something about provisioning.
- Something about CI.
- Uses Skaffold and ksync in the background.
- Has build packs that generate config files similarly to Draft! 😊

garden

Build & Deploy

- Draft
- Garden
- Jenkins X
- **Skaffold**
- Tilt

garden

Notes:

- Automatically builds & deploys!
- For those already familiar with k8s, it doesn't require learning new concepts.
- Requires k8s manifests! Not for beginners.
- Easy set up if you already have them.
- No support for dependencies, tests, or bootstrapping.

SKAFFOLD

Build & Deploy

- Draft
- Garden
- Jenkins X
- **Skaffold**
- Tilt

garden

```
→  getting-started (skaffold-master) skaffold dev --default-repo master
Generating tags...
 - master/gcr_io_k8s-skaffold_skaffold-example -> master/gcr_io_k8s-skaffold_skaffold-
example:v0.32.0-28-g6bd1d50a
Tags generated in 15.581571ms
Starting build...
Found [minikube] context, using local docker daemon.
Building [master/gcr_io_k8s-skaffold_skaffold-example]...
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM golang:1.12.5-alpine3.9 as builder
 ---> c7330979841b
Step 2/6 : COPY main.go .
 ---> Using cache
 ---> 4a407d68d98b
Step 3/6 : RUN go build -o /app main.go
 ---> Using cache
 ---> 9a5ab26344e5
Step 4/6 : FROM alpine:3.9
 ---> 055936d39205
Step 5/6 : CMD ["./app"]
 ---> Using cache
 ---> a70bbd56d26a
Step 6/6 : COPY --from=builder /app .
 ---> Using cache
 ---> dc6de832b819
Successfully built dc6de832b819
Successfully tagged master/gcr_io_k8s-skaffold_skaffold-example:v0.32.0-28-g6bd1d50a
Build complete in 61.458378ms
Starting test...
Test complete in 5.089µs
Starting deploy...
kubectl client version: 1.14
pod/getting-started created
Deploy complete in 320.901518ms
Watching for changes every 1s...
[getting-started] Hello world!
[getting-started] Hello world!
[getting-started] Hello world!
```

Build & Deploy

- Draft
- Garden
- Jenkins X
- **Skaffold**
- Tilt

garden

```
[getting-started] Hello world!
[getting-started] Hello world!
Generating tags...
 - master/gcr_io_k8s-skaffold_skaffold-example -> master/gcr_io_k8s-skaffold_skaffold-
example:v0.32.0-28-g6bd1d50a-dirty
Tags generated in 45.095825ms
Starting build...
Found [minikube] context, using local docker daemon.
Building [master/gcr_io_k8s-skaffold_skaffold-example]...
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM golang:1.12.5-alpine3.9 as builder
 ---> c7330979841b
Step 2/6 : COPY main.go .
 ---> 6ec58eeadbdb
Step 3/6 : RUN go build -o /app main.go
 ---> Running in 154f95e0cab2
Removing intermediate container 154f95e0cab2
 ---> 7cb3b1cd4e40
Step 4/6 : FROM alpine:3.9
 ---> 055936d39205
Step 5/6 : CMD ["./app"]
 ---> Using cache
 ---> a70bbd56d26a
Step 6/6 : COPY --from=builder /app .
 ---> 5f7022e1c5a3
Successfully built 5f7022e1c5a3
Successfully tagged master/gcr_io_k8s-skaffold_skaffold-example:v0.32.0-28-g6bd1d50a-d
irty
Build complete in 1.222589614s
Starting test...
Test complete in 4.015µs
Starting deploy...
kubectl client version: 1.14
pod/getting-started configured
Deploy complete in 253.558502ms
Watching for changes every 1s...
[getting-started] Hello world!!
[getting-started] Hello world!!
```

Build & Deploy

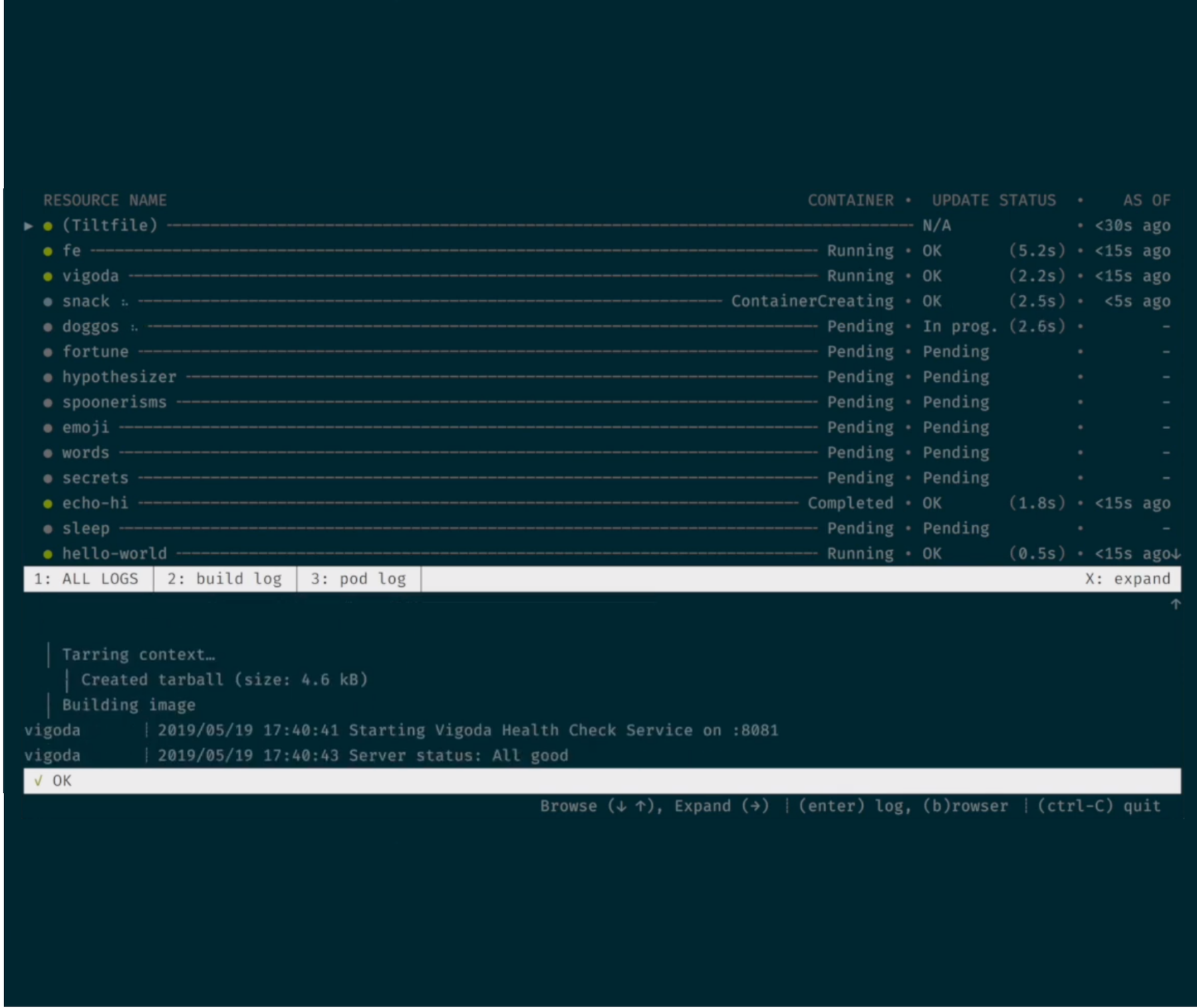
- Draft
- Garden
- Jenkins X
- Skaffold
- **Tilt**

Notes:

- Like Skaffold: Requires k8s manifests; easy set up if you already have them.
- Gorgeous UI, dashboard, and logging.
- Vs. Skaffold? Skaffold's concepts map almost directly into Tilt.
- Starlark configs!

Build & Deploy

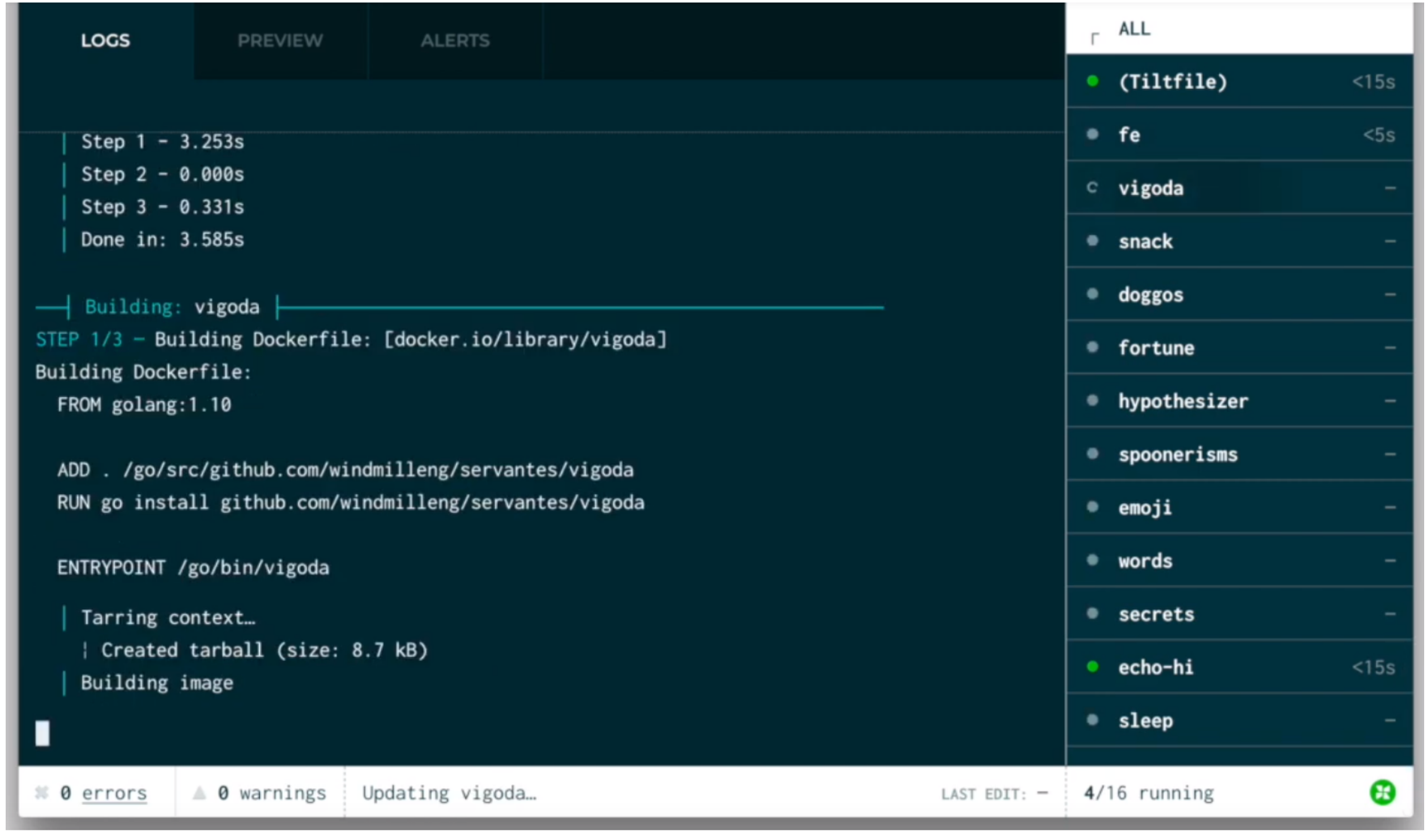
- Draft
- Garden
- Jenkins X
- Skaffold
- **Tilt**



garden

Build & Deploy

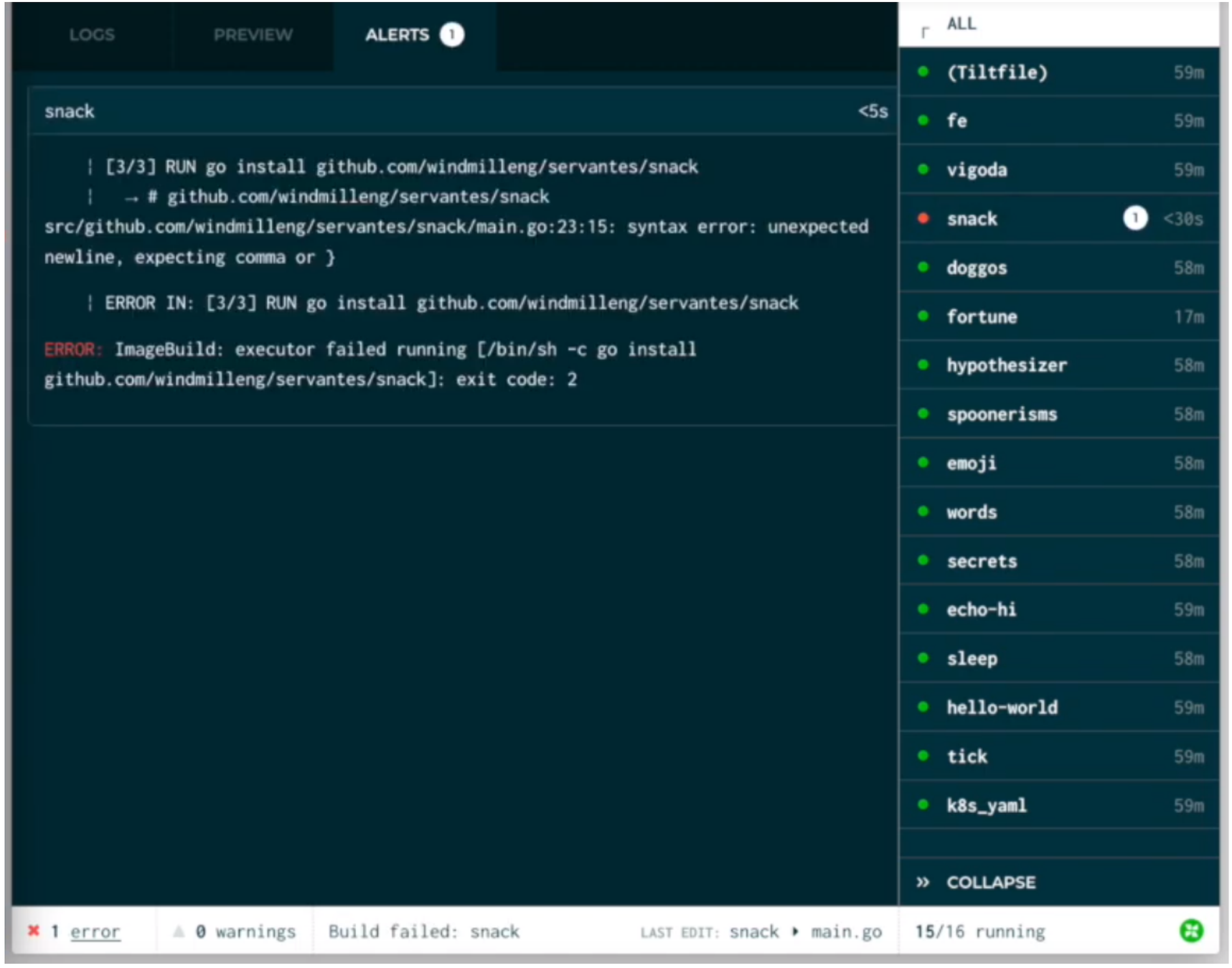
- Draft
- Garden
- Jenkins X
- Skaffold
- **Tilt**



garden

Build & Deploy

- Draft
- Garden
- Jenkins X
- Skaffold
- **Tilt**



garden

Toolchains:

Omnipresent:

- Helm
- kubectl

Connect:

- Forge ☠️
- inlets
- ksync
- kubefwd
- Okteto
- Telepresence

Build&Deploy Cycle:

- Draft ☠️
- Garden
- Jenkins X
- Skaffold
- Tilt

Debug:

- kubefwd
- Squash
- Stern

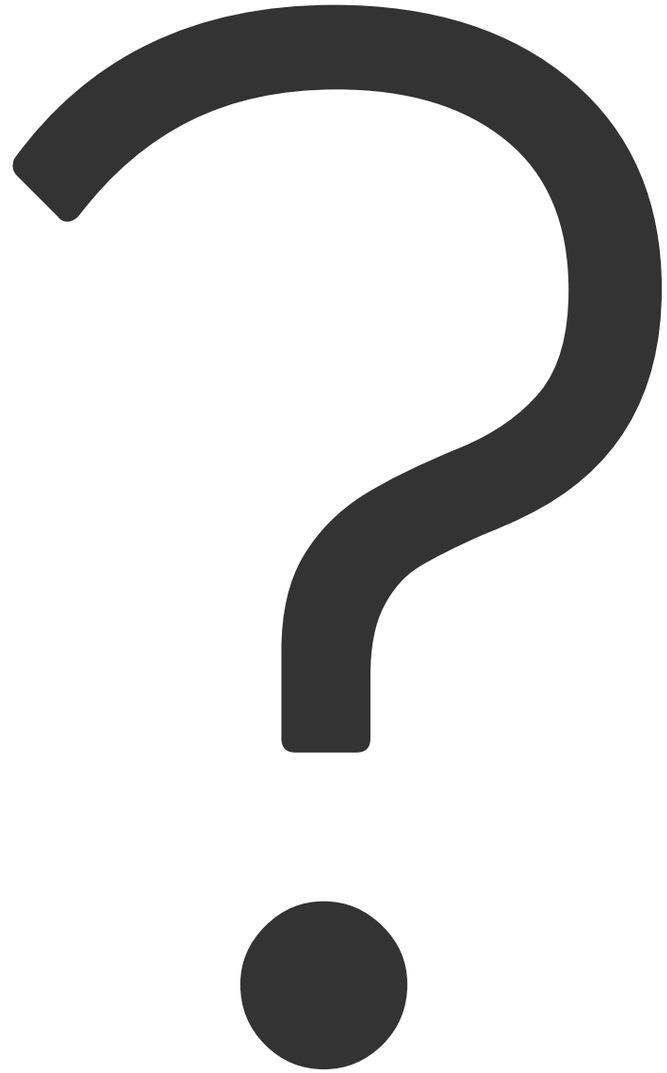
garden

“It works on my cluster...”

@maeddes

Still missing:

- Better docs!
- Consistent naming for things across different tools.
- Better support for tests!
- `(ls *.go | entr -r ./mytest.sh)`
- Beginner friendliness.
- **Clarity. A cohesive story.**



Links:

- Draft → draft.sh
- entr → eradman.com/entrproject/
- Forge → forge.sh
- Garden → garden.io
- Helm → helm.sh
- inlets → github.com/inlets/inlets & inlets-operator
- Jenkins X → jenkins-x.io
- ksync → github.com/vapor-ware/ksync
- kubectl 'Debug Services' → kubernetes.io/docs/tasks/debug-application-cluster/debug-service/
- kubefwd → kubefwd.com
- Okteto → okteto.com
- Skaffold → skaffold.dev
- Squash → github.com/solo-io/squash
- Stern → github.com/wercker/stern
- Telepresence → www.telepresence.io
- Tilt → tilt.dev

> You can't have a clusterf!$& without a cluster.
>
> @krisnova

ellen körbes

- developer relations
- ellen@garden.io
- @ellenkorbes
- #garden@slack.k8s.io
- they/them

garden

garden

ellen@garden.io | @ellenkorbes

Please

Remember to rate this session

Thank you!

@GOTOber

gotober.com