EXPLORING KUBERNETES

JAMES DABBS @ JAVASCRIPTLA

OUTLINE

- BACKGROUND
- EXPLORATIONS WITH LENS
- ITERATING WITH SKAFFOLD
- DEFINING COMPONENTS WITH CDK8S
- SUMMARY, TAKEAWAYS & QUESTIONS

BACKGROUND: ME

- APP DEV STUMBLING DOWN THE INFRASTRUCTURE GRADIENT
  - DABBLED WITH INFRASTRUCTURE-AS-CODE
- FRUSTRATED BY SLOW, FLAKY BUILDS AND LONG ITERATIONS

BACKGROUND: ME

- APPLICATION INFRASTRUCTURE @ PROCORE
- DEVELOP DEV-FACING ABSTRACTIONS FOR OUR INTERNAL PLATFORM
- GOAL: EMPOWER DEVS TO SELF-SERVE WITH MINIMAL COGNITIVE LOAD

I've learned a mix of tools and techniques that have made this world approachable. I hope to share both in this talk.

HOW I LEARNED TO STOP WORRYING AND LOVE INFRASTRUCTURE DEV

BACKGROUND: KUBERNETES (K8S)

- CONTAINER ORCHESTRATION PLATFORM FROM GOOGLE
  - EVERYTHING IS DEFINED AS STRUCTURED YAML*
    - SO HOT RIGHT NOW

We'll see some k8s manifests shortly.

^ One of the key things about k8s is that it is declarative - you operate entirely by reading and writing YAML files with a defined (if abstruse) schema.

HELLO K8S

DEPLOYMENT

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

SERVICE

```yaml
apiVersion: v1
kind: Service
metadata:
  name: schedule
spec:
  type: ClusterIP
  ports:
    - port: 3000
      targetPort: 80
  selector:
    app: nginx
```

FRIDAY

MANAGE MEETINGS WITH

- A RECURRING GOOGLE CALENDAR INVITE
- MEETING NOTES FOR EACH WEEK, GENERATED FROM A TEMPLATE AND LINKED IN THE CALENDAR INVITE
- A ROTATING SCRIBE AND EMCEE AMONGST THE MEETING MEMBERS
- SLACK REMINDERS SENT OUT MORNING-OF WITH REMINDERS

FRIDAY - NESTJS SERVICES

- `schedule` - HTTP SERVICE WITH GOOGLE CALENDAR INTEGRATION
- `minutes` - HTTP SERVICE USING CONFLUENCE'S API
- `slack` - LISTENS ON A KAFKA TOPIC, POSTS TO SLACK
- `meetings` - COORDINATING SERVICE USING PRISMA

LENS

& AN EXPLORATION OF KUBERNETES

Live demo through
^ - Namespaces
^ - Cluster ingress
^ - Strimzi Kafka operator
^ - Application deployed in the Friday namespace
^ - Explore Services => Deployments => Pods
^ - Look at manifests for each
^ - Kafka?

LENS

- GUI FOR KUBERNETES
- GREAT FOR EXPLORING YOUR CLUSTER OR `k8s` BROADLY
- HELPFUL FOR DEBUGGING `k8s` APPLICATIONS

SKAFFOLD & MAKING CHANGES

Live demo through
^ - skaffold dev
^ - skaffold.yaml file
^ - sync definitions
^ - infra.k8s.yaml
^ - curl meetings.localhost
^ - trace ingress => service => deployment => image entrypoint => npm run start:dev
^ - make update in meetings/application.controller

APP DEV LOOP

- MAKE CHANGE
- `skaffold` SYNCS FILES TO CONTAINER
- WEBPACK HMR PICKS UP CHANGES

TTL: FAST™ (<5S)

LIBRARY DEV LOOP

- MAKE CHANGE AND BUMP VERSION
  - `npm run publish`
  - UPDATE VERSION IN CONSUMER AND SAVE
- `skaffold` REBUILDS IMAGE, INSTALLING NEW VERSION

TTL: COULD BE BETTER (~75S)

*_But_ is this something you need to optimize?

^ My normal flow here is to TDD library work with `jest --watchAll`

^ until it's ready for an `-rc#` version

SKAFFOLD

- QUICKLY SYNC CODE AND MANIFEST CHANGES TO A `k8s` CLUSTER
  - MORE THAN JUST `skaffold dev`
    - SUPPORTS DIFFERENT PROFILES

Note that all of the example code here is optimized for application development

^ - `Dockerfiles` include dev dependencies and run `start:dev` to watch for file changes

^ - `Dockerfiles` install libraries from a (private) NPM repo

^ Could define other profiles for e.g. building `prod`-suitable images or for faster iteration on libraries

CDK8S & DEFINING K8S COMPONENTS

Explore

^ - main chart, subcharts, generated submanifests & full manifest

^ - note that we can import custom resource definitions like for the Kafka operator

^ - example: refactor to add a health check to schedule sim. meetings

COMPONENT DEV LOOP

› MAKE CHANGE

› `start:dev` TYPECHECKS AND RENDERS MANIFEST

› `skaffold dev` DEPLOYS IT

TTL: NOT BAD (~30S)*

*Depending wildly on the nature of the change for the deploy to stabilize

^ Almost all time is spent in the k8s deploy rollout

CDK8S

- BRING YOUR USUAL TOOLS TO BEAR TO ENCAPSULATE. REUSE. AND VERIFY LOGIC
- TYPESCRIPT STANDARD AUTOCOMPLETE. HINTING. AND TYPECHECKING
- SHARE BUSINESS LOGIC (E.G. NAMING CONVENTIONS) BETWEEN APP AND INFRA LAYERS

CDK8S

SEE ALSO

- PYTHON AND JAVA IMPLEMENTATIONS OF CDK8S
- CDKTF FOR TERRAFORM

RECAP & TAKEAWAYS

SOME TOOLS

- LENS
- SKAFFOLD
- CDK8S

RECAP & TAKEAWAYS

TOOLS WHICH IMPROVE DISCOVERABILITY ARE KEY. ESPECIALLY AS YOU'RE BUILDING YOUR MENTAL MAP

RECAP & TAKEAWAYS

- OPTIMIZE YOUR FEEDBACK LOOPS EARLY AND OFTEN
  - FASTER FEEDBACK = FASTER LEARNING

QUESTIONS?

EXPLORING KUBERNETES

JAMES DABBS @ JAVASCRIPTLA