INTEGRATING DEVELOPER AND OPERATOR EXPERIENCE IN K8S

BRENDA BURNS

VELOCITY – NYC - 2018
FIRST A QUESTION?

• Do you love your development environment?
FIRST A QUESTION? (OR TWO)

• Do you love your development environment?
• Does your development environment love you?
WHAT DO I MEAN?

• We're editing code in the cloud
• We're building code in the cloud
• We're deploying code in the cloud
• We're debugging code in the cloud
WHAT DO I MEAN?

• But our tools are tied to a single machine
• And often our tools are confined to a single persona
GOALS AND NON-GOALS

Goals:
• Make developing, testing, deploying and debugging seamless
• Build modular, composable tooling experiences

Non-goals:
• Develop an all-encompassing "railed" experience
• Assuming that everyone will use these tools (all or some)
A QUICK WORD FROM OUR SPONSORS...

• J/K
THE 'I' IN 'IDE' IS FOR INTEGRATION

• Code doesn't run in a vacuum

• Most of the things our code 'integrates' with don't live on our machine

• Some of this we're somewhat good at:
  • Git

• Most of it, we're pretty bad at:
  • Services, Replication, Logs, ....

• And even the things that are on the machine often aren't great...
GETTING STARTED (WITH K8S)

Code → Compiler → Dockerfile → Docker → Kubernetes YAML → kubectl

Binary → Dockerfile → Image → Service
GETTING STARTED (WITH K8S)

Compiler

Docker

kubectl

But tools can help!
FIRST THINGS FIRST, A PLACE TO DEPLOY...

• The deployment environment for code used to be implicit.
• Not anymore.
• But tools can help with that too...
AND THEN THERE'S THAT WHOLE DOCKERFILE AND YAML THING...

• Three files
• Three tools
• Three syntaxes

• Just to deploy one app!
DRAFT DEMO

• https://draft.sh
BUILDING MODERN TOOLS
BUILDING MODERN TOOLS
BUILDING MODERN TOOLS
BUILDING AND DEPLOYING IMAGES
BUILDING AND DEPLOYING WITH DRAFT
VISIBILITY (OR THE LACK THERE-OF)
EXPLORING YOUR CLUSTER

• Connect the development and deployment environment together
• Browse your cluster
• Visualize service health
• Don't learn 'awk' (yet)
CONNECTING TO YOUR APPLICATION

• My code's in the cloud... Now what?
EXPLORING YOUR PODS (AND CONTAINERS)

• Integrating Pod logs

• Integrating Pod terminal

```
apiVersion: v1
kind: Pod
metadata:
  name: foo
...

Hello world!
Request handled
...
```
DEBUGGING YOUR CODE

• Bringing your services to a local proxy
DEBUGGING YOUR CODE

- Synchronizing external state to your local repository

```go
func main() {
    http.ListenAndServe(:80)
}
```
UPGRADING YOUR JOBS

• Edit and visual diff.
UPGRADING YOUR JOBS

• Making the changes in a principled way
LOOKING FORWARD

• Cloud state and cloud experiences.
• Merging logs from many different sources...

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: foo
...
```

```json
{ app: frontend, stage: production }
```
LOOKING FORWARD

Cloud-idiomatic code
const server = http.createServer((request, response) => {
    console.log(request.url);
    response.end(`Hello World: hostname: ${os.hostname()}
`);
});

mp.containerize(
    { repository: 'docker.io/docker-user-goes-here', },
) => { server.listen(port,
    (err) => {
        if (err) { return console.log('server startup error: ', err); } 
        console.log(`server up on ${port}`);
    });
};
from metaparticle import containerize
class MyHandler(SimpleHTTPServer.SimpleHTTPRequestHandler):
    ...

@containerize(
    'docker.io/your-docker-user-goes-here', options={'name': 'my-image', 'publish': True})

def main():
    Handler = MyHandler
    httpd = socketserver.TCPServer(('', port), Handler) httpd.serve_forever()
    if __name__ == '__main__':
        main()
CONTRIBUTIONS NEEDED

• What about monitoring?
• What about collaboration?
• Configuration languages?

• https://github.com/Azure/vscode-kubernetes-tools
• https://github.com/helm/helm