KubeCon | CloudNativeCon

North America 2017

Local Dev w/ K8s

Ryan Jarvinen, Developer Advocate, Red Hat
@RyanJ

http://bit.ly/kubecon-dev

presented by @ryanj, Developer Advocate at Red Hat

We Are Terrible at Pitching Kubernetes to Developers

Why?

Kubernetes

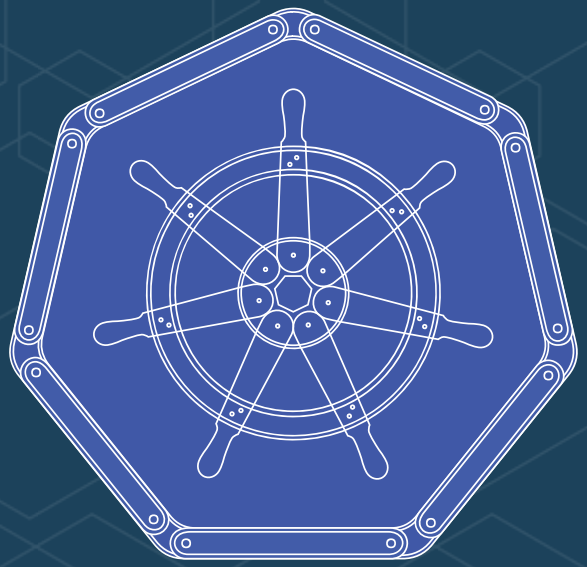
(an ops tool)


CHEF CRAFT
Weekly Pill Organizer
Organizador semanal de tabletas
Seven Day Supply


CHEF CRAFT
Weekly Pill Organizer
Organizador semanal de tabletas
Seven Day Supply

When used as directed, provides relief for the following:

1. standardized terminology & packaging - containers, volumes, podspecs, charts
2. load balancing - services
3. scaling automation - replica sets
4. delivery automation - deployments
5. high availability - automated health checking and replacement
6. distributed scheduling and resource management - RBAC, namespaces, labels, federation
7. ???



meanwhile...

What is an App?

1. ~~repo code~~
2. ~~docker image~~
3. kubernetes spec files
4. charts
5. kubectl get all -l app=myapp -n mynamespace

Proposal: Label Recommendations

How should we be talking to Developers about Kubernetes?

Q: Why Kubernetes?

A: Development Velocity


WILLIE
JANIS
TOWNES
STEVIE RAY
ROKY
AUSTIN
Live Music
CAPITAL
of the WORLD!
TEXAS

A Case Study: Enterprise Records, Inc.

The Ops team has heard great things about Kubernetes, and is interested in giving it a try - but they're having difficulty convincing other teams of the value

Product team needs:

More

(always more)

The web team is confused by all the new terminology, and is under a lot of pressure to focus on delivering new tracks to customers


Don’t blow this for us Gene!

Convincing the team (minimal onboarding):

1. Getting started is easy
2. Share what you know (and model your I/O)
3. Choose the right toolchain

1. The Easy Part

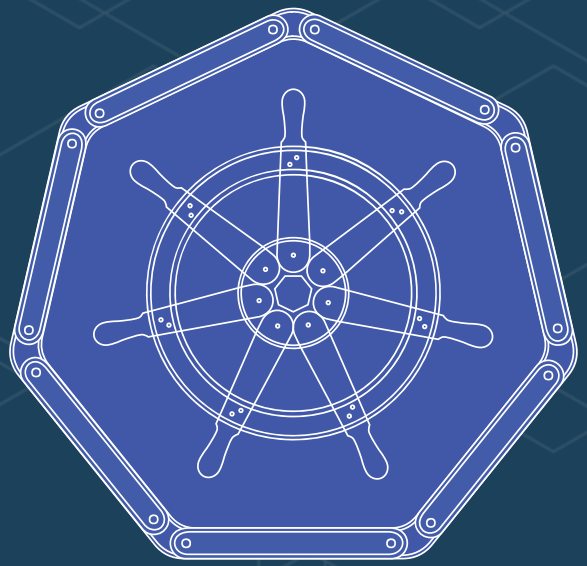
is

```
minikube start
```

- Staging down?
- Ops not Ready?

  No Excuses!

!!Everyone get a K8s!!



Minikube

- Minikube Docs
- bit.ly/k8s-minikube

2. Share What You Know

and model your I/O

Share What You Know `--dry-run`

Generate kubernetes `deployment` and `service` specifications, both named `metrics-review`:

```
kubectl run metrics-review --image=quay.io/ryanj/metrics-k8s \
--expose --port=2015 --service-overrides='{ "spec": { "type": "NodePort" } }'
--dry-run -o yaml > metrics-review.yaml
```

Share What You Know `--dry-run`

Test your generated spec:

```
kubectl create -f metrics-review.yaml
```

Minikube users will be able to open the resulting service in their browser by running:

```
minikube service metrics-review
```

Model Your I/O

Example Repo

Create a local clone of this `metrics-k8s` repo:

```
git clone http://github.com/ryanj/metrics-k8s
```

Preview - local files

Next, share your local repo contents with minikube:

```
minikube mount $(pwd):/var/www/html
```

Preview - hostPath

Then, produce a new deployment spec that includes (minimal) support for live development workflows:

1. `cp metrics-review.yaml metrics-dev.yaml`
2. replace `metrics-review` with `metrics-dev` (global)
3. Add a `hostPort` volume to access your local repo:

```
      spec:
        containers:
        - image: quay.io/ryanj/metrics-k8s
          name: metrics-dev
          ports:
          - containerPort: 2015
          resources: {}
+         volumeMounts:
+         - mountPath: /var/www/html
+           name: metrics-src
+       volumes:
+       - name: metrics-src
+         hostPath:
+           path: /var/www/html
  status: {}
```

Share what you know

The resulting file should look just like the included metrics-dev.yaml file from the `metrics-k8s` git repo.

Try launching it with:

```
kubectl create -f metrics-dev.yaml
```

Share what you know - Rollout Testing

Eval this

```
minikube docker-env
```

to send newly-built images to minikube's docker daemon:

```
docker build .
```

3. The Hard Part

Keeping it simple, and choosing the right tools for the job

The future is already here — it's just not very evenly distributed.
(W.Gibson)

Typical container adoption path:

1. docker
2. volumes, PVs
3. minikube
4. k8s modeling and scalability via spec files, pods, and other abstractions
5. charts, openshift templates, or hand-rolled manifest / spec templating
6. monocular, kubeapps, ServiceCatalog
7. PaaS?

Draft

Make it easy to get started

Charts

Share what you know

Helm & Tiller

Share more

Brigade and Kashti

Do more

Telepresence

Access more

minishift and oc

Security Enhanced Kubernetes

Easy, right?

More Learning Opportunities

1. Kubernetes.io Tutorials
   https://kubernetes.io/docs/tutorials/
2. Katacoda https://katacoda.com/courses/kubernetes
3. RyanJ's K8s-workshops http://bit.ly/k8s-workshops
4. Interactive learning for OpenShift:
   http://learn.openshift.com

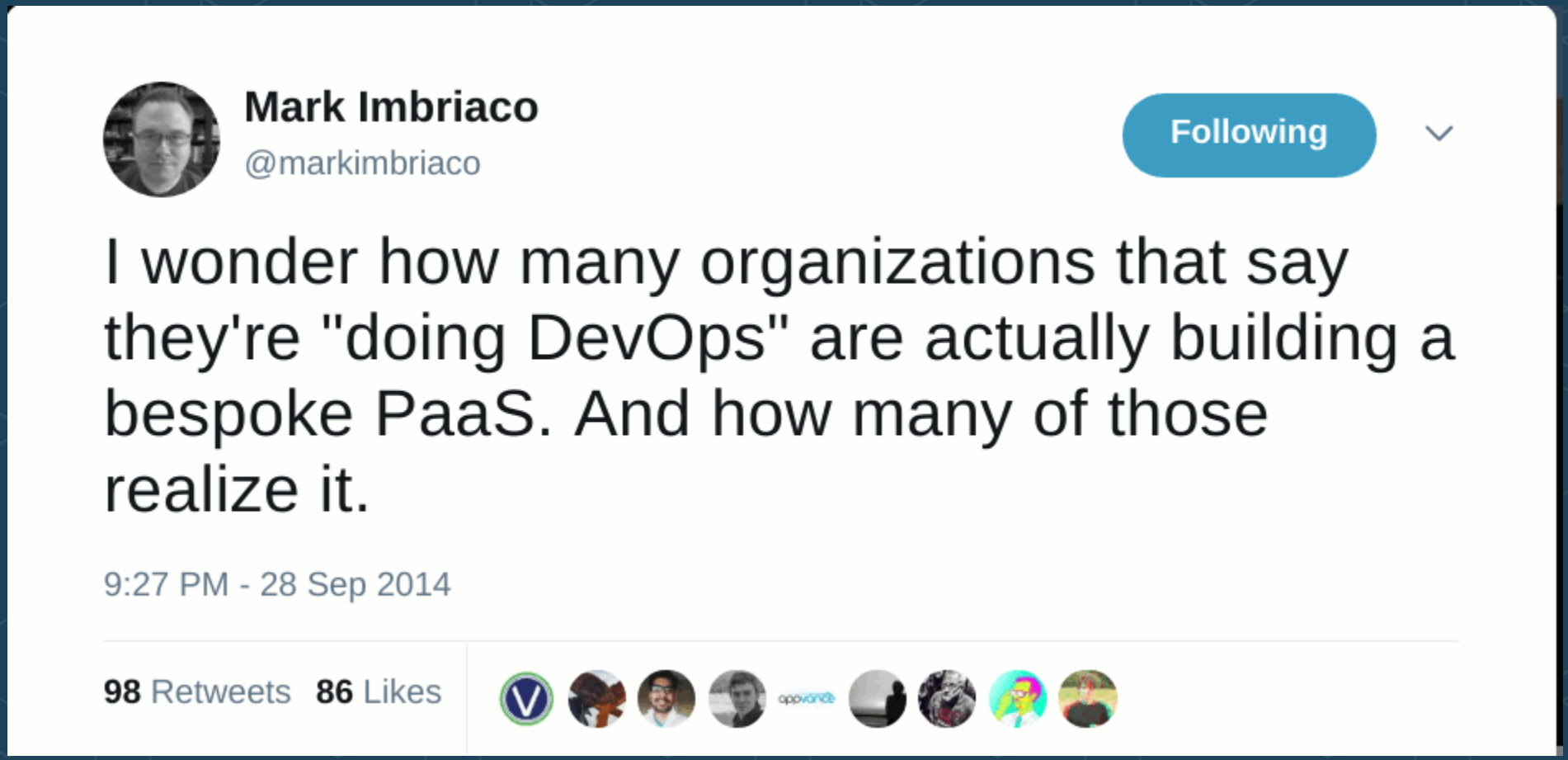
Include the whole team:

- Developers: Want to get ahead? Model your I/O, and Share What You Know!
- Architects: Figure out who owns manifest creation, maintanence, and distribution
- QA folks: look forward to saying: "can't repro - works fine on my Kubernetes"
- Ops: provide cloud resources grants to teams, make sure prod has enough IaaS, ensure platform uptime, upgrades, logging, and metrics
- Security & Compliance: RBAC, config and secrets management; Secret rotation policies; Monitor for CVEs and apply security patches from upstream

Join the community on Slack in #kubernetes-users, and in #SIG-Apps!

Share What You Know: Help us develop a range of solutions that expose and/or hide kubernetes in appropriate ways

Learn to deliver consistently using containers



Choose the right tools for the job

then get back to making gold records

Thank You!

@RyanJ

bit.ly/kubecon-dev