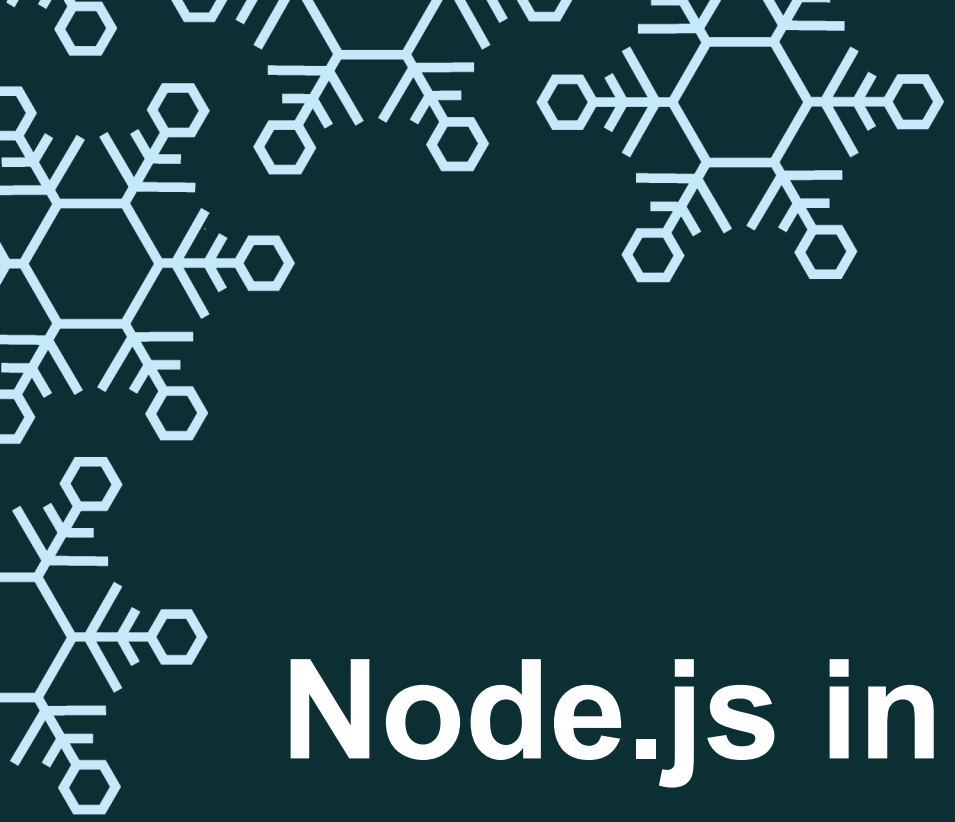
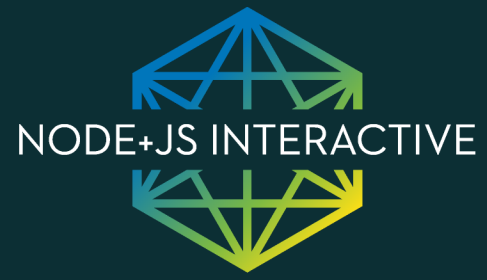




December 11–12, 2019
Montreal, Canada

Node.js in a Kubernetes World

Michael Dawson, Community Lead for Node.js, IBM

About: Michael Dawson

IBM Community Lead for Node.js

- Active Node.js community member
  - Collaborator
  - Node.js Technical Steering Committee TSC Chair
  - Community Committee member
  - Node.js OpenJS Foundation Board Member
  - Working group(s) member/leadership

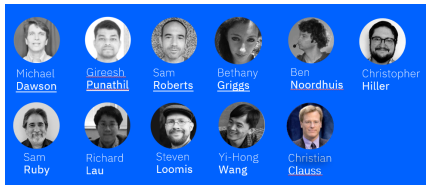
- Twitter: @mhdawson1
- GitHub: @mhdawson
- Linkedin: https://www.linkedin.com/in/michael-dawson-6051282

Agenda

- IBM’s JavaScript/Node.js Focus
- Cloud Native Development
- Key Cloud Native Development Tasks for the Node.js Developer
- Separating Concerns
- Available Tools
- Q/A

IBM's JavaScript/Node.js Focus

- Community Leadership



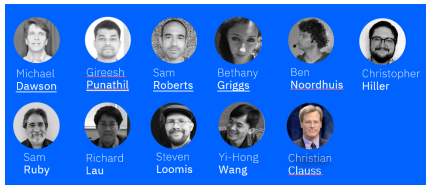
- IBM Platform Enablement

- Cloud Enablement

- Developer Enablement
- Commercial Support

IBM's JavaScript/Node.js Focus

- Community Leadership



- IBM Platform Enablement

- Cloud Enablement

- **Developer Enablement**
- Commercial Support

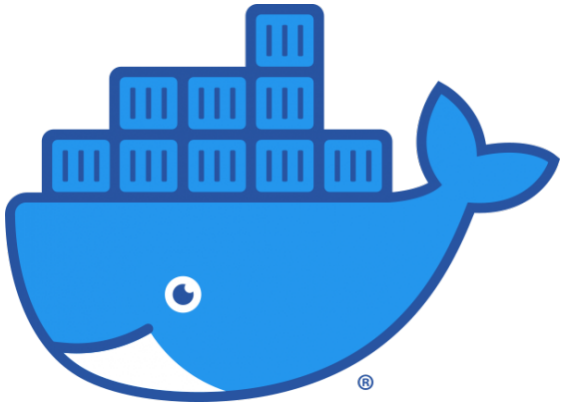
Developer Enablement

- Rapid Development
- Successful Production Deployments

Developer Enablement

- Rapid Development

- Successful Production Deployments

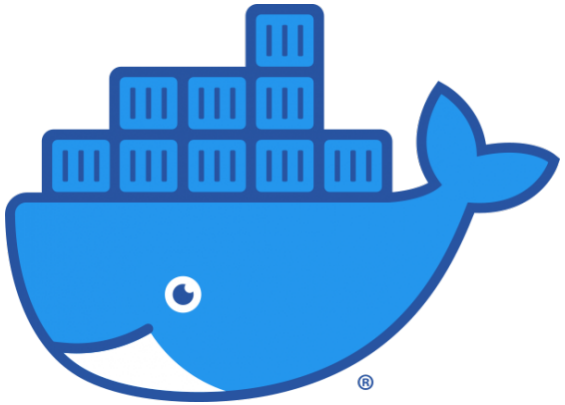


https://www.docker.com/

Developer Enablement

- Rapid Development

- Successful Production Deployments



https://www.docker.com/

https://github.com/kubernetes/kubernetes

Cloud Native Development

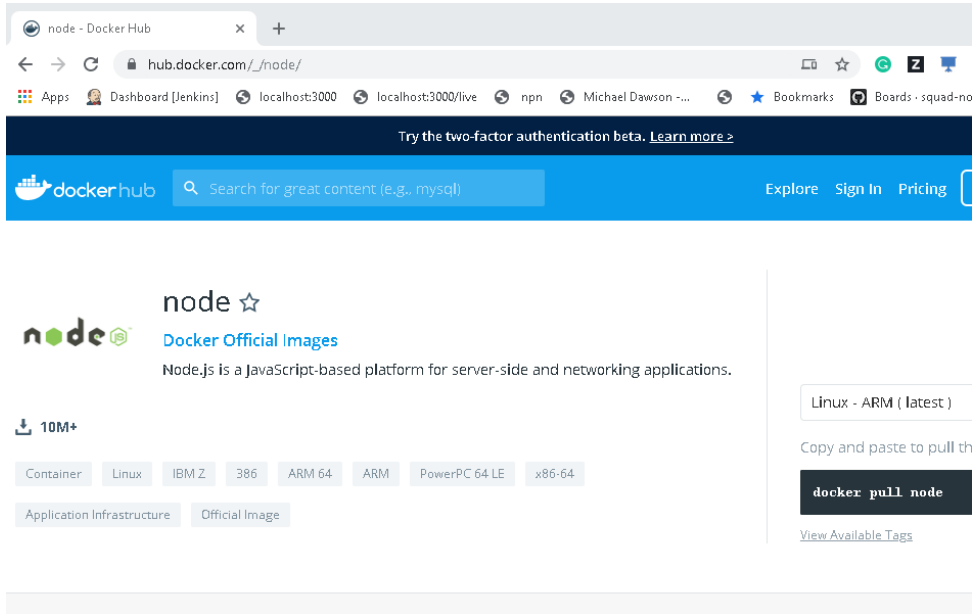
- 12 Factor Applications - https://12factor.net/
  - Keep development, staging, and production as similar as possible
  - Treat Logs as event streams
  - Export services via port binding

- Some key Tasks
  - Building Docker Images
  - Deployment
  - Testing
  - Health checking
  - Logs
  - Metrics
  - Upgrades

Building Docker Images – Base Images

- Community Node.js Images (10M+ downloads)
  - Ex node:12.13.1
  - Add Node.js binaries to existing distro, in default path
  - Variants
    - node:<version>
    - Node:<version>-slim – minimal packages to run node
    - <version>-alpine – smaller container

https://github.com/nodejs/docker-node/blob/master/docs/BestPractices.md

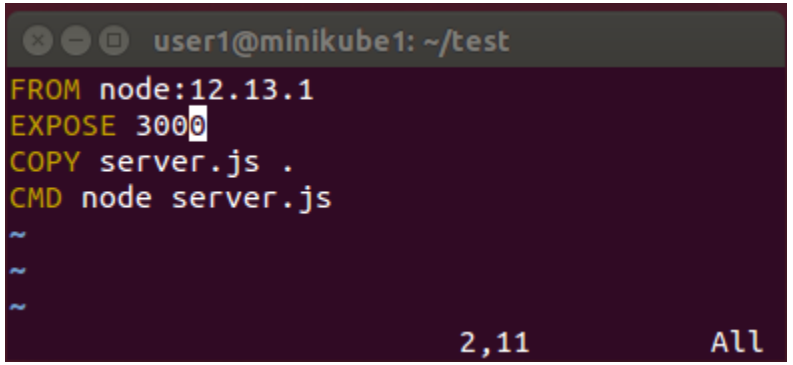


https://hub.docker.com/_/node/

- Others:
  - registry.access.redhat.com/ubi8/nodejs-10

Building Docker Images - Building

- Dockerfile



- Building
  - docker build –t test .
  - Use Multi-stage Build to minimize size
    - For example: native addons

    https://docs.docker.com/develop/develop-images/multistage-build/

- Push to registry

- Run/test under docker/kubernetes

Deployment Artifacts

- **YAML**

  kubectl apply –f deployment.yaml
  kubectl apply –f service.yaml
  kubectl apply –f ingress.yaml

- Helm Chart

- Operators

Deployment Artifacts

- YAML

- **Helm Charts** https://github.com/helm/charts
  - **deployment.yaml**
  - **service.yaml**
  - **ingeress.yaml**

- Operators

```
Package-name/
  charts/
  templates/
  Chart.yaml
  LICENSE
  README.md
  requirements.yaml
  values.yaml
```

Deployment Artifacts

- YAML
- Helm Charts
- **Operators** https://github.com/operator-framework

Testing

- Keep development, staging, and production as similar as possible
- Overhead ?
  - Docker build, start container and access Docker image to test each change!
  - Delete Pods, Re-deploy in Kubernetes

Liveness and Readiness

- Liveness – when to restart container
- Readiness – when ready to start accepting traffic

Liveness

- Probe Types

  Run a command in the container
  HTTP probe
  TCP probe

- http probe common for Node.js Deployments
  - OK >= 200 < 400

- Can/should check more than web server
  - Separate endpoint

```
containers:
- name: test1
  image: test1:latest
  args:
  - /server
  livenessProbe:
    httpGet:
      path: /live
      port: 3000
    initialDelaySeconds: 5
    periodSeconds: 10
```

Liveness

- Probe Types

  Run a command in the container
  HTTP probe
  TCP probe

- http probe common for Node.js Deployments
  - OK >= 200 < 400

- Can/should check more than web server

```
containers:
- name: test1
  image: test1:latest
  args:
  - /server
  livenessProbe:
    httpGet:
      path: /live
      port: 3000
    initialDelaySeconds: 5
    periodSeconds: 10
```

Readiness

- Same options/config as liveness probe
- No restart but traffic is not routed to instance

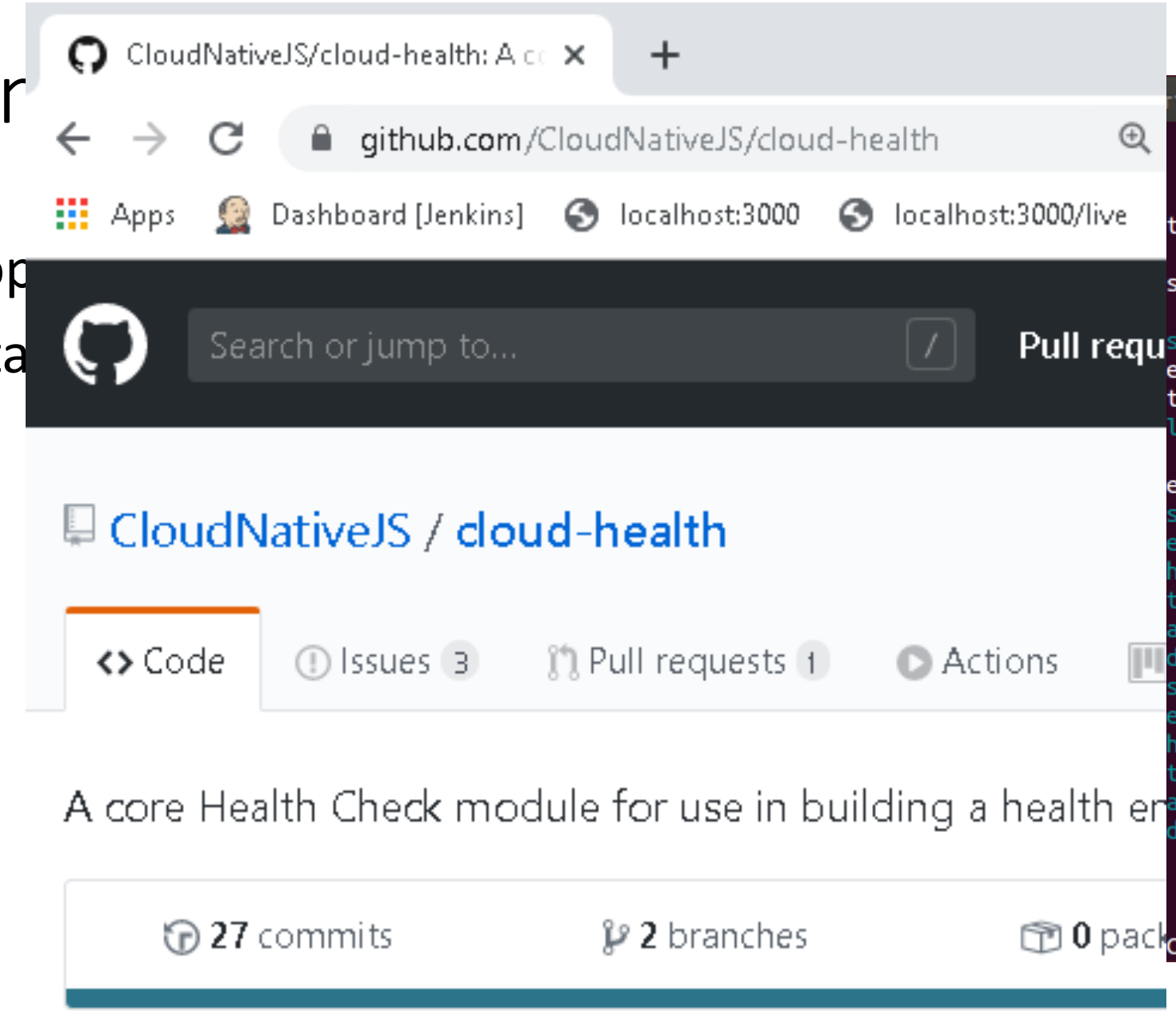
```
user1@minikube1:~/test$ kubectl get all
NAME          READY   STATUS    RESTARTS   AGE
pod/test1     0/1     Running   0          6s
```

user1@minikube1: ~/test

```
apiVersion: v1
kind: Pod
metadata:
  name: test1
  labels:
    app: test1
spec:
  containers:
  - name: test1
    image: test1:latest
    imagePullPolicy: Never
    args:
    - /server
    livenessProbe:
      httpGet:
        path: /live
        port: 3000
      initialDelaySeconds: 5
      periodSeconds: 10
    readinessProbe:
      httpGet:
        path: /ready
        port: 3000
      initialDelaySeconds: 1
      periodSeconds: 10
~
~
~
<" 25L, 439C                 20,13         All
```

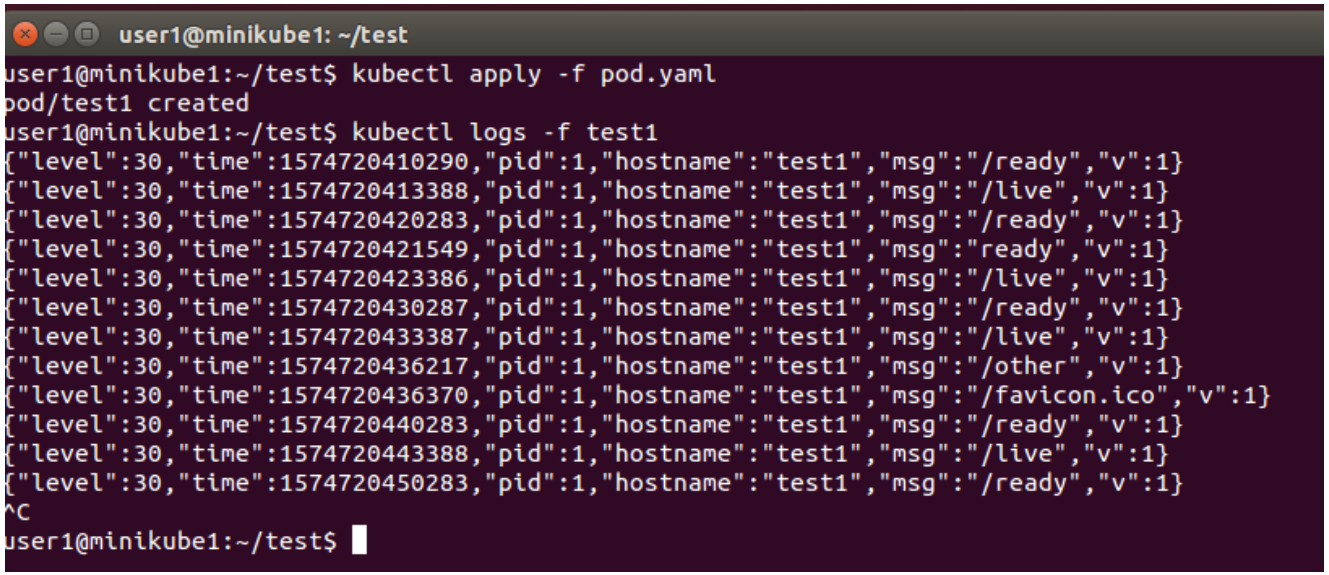
Readir

- Same op
- No resta



Logs

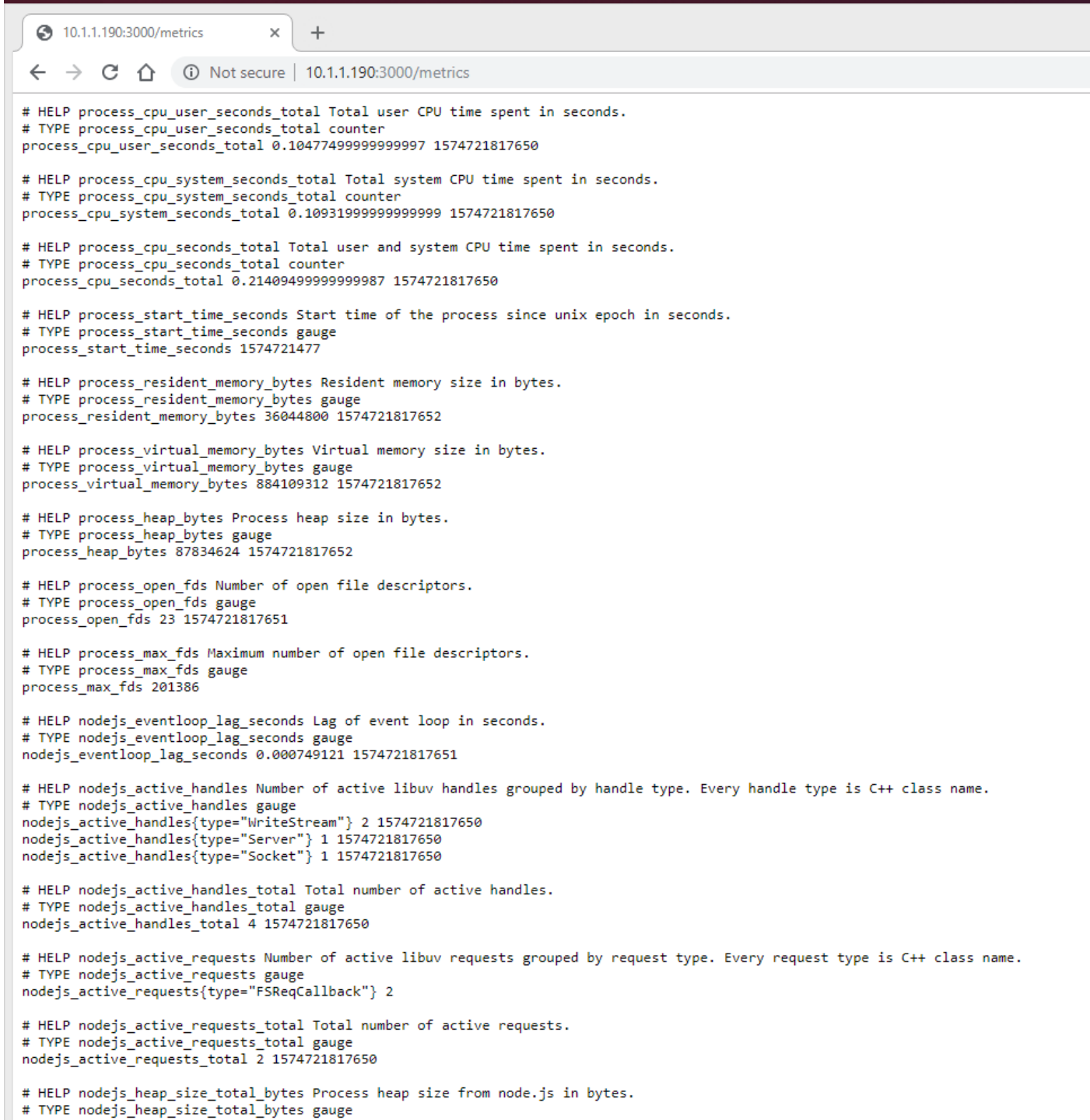
- Containers are ephemeral
- Log to stdout
- Structured logging is the trend



kubectl logs –f pod test1

Metrics

- Prometheus https://prometheus.io/
- Expose /metrics endpoint to return metrics
  - prom-client
  - appmetrics-Prometheus
- Default and Application specific metrics




```
# HELP process_cpu_user_seconds_total Total user CPU time spent in seconds.
# TYPE process_cpu_user_seconds_total counter
process_cpu_user_seconds_total 0.10477499999999997 1574721817650

# HELP process_cpu_system_seconds_total Total system CPU time spent in seconds.
# TYPE process_cpu_system_seconds_total counter
process_cpu_system_seconds_total 0.10931999999999999 1574721817650

# HELP process_cpu_seconds_total Total user and system CPU time spent in seconds.
# TYPE process_cpu_seconds_total counter
process_cpu_seconds_total 0.21409499999999987 1574721817650

# HELP process_start_time_seconds Start time of the process since unix epoch in seconds.
# TYPE process_start_time_seconds gauge
process_start_time_seconds 1574721477

# HELP process_resident_memory_bytes Resident memory size in bytes.
# TYPE process_resident_memory_bytes gauge
process_resident_memory_bytes 36044800 1574721817652

# HELP process_virtual_memory_bytes Virtual memory size in bytes.
# TYPE process_virtual_memory_bytes gauge
process_virtual_memory_bytes 884109312 1574721817652

# HELP process_heap_bytes Process heap size in bytes.
# TYPE process_heap_bytes gauge
process_heap_bytes 87834624 1574721817652

# HELP process_open_fds Number of open file descriptors.
# TYPE process_open_fds gauge
process_open_fds 23 1574721817651

# HELP process_max_fds Maximum number of open file descriptors.
# TYPE process_max_fds gauge
process_max_fds 201386

# HELP nodejs_eventloop_lag_seconds Lag of event loop in seconds.
# TYPE nodejs_eventloop_lag_seconds gauge
nodejs_eventloop_lag_seconds 0.000749121 1574721817651

# HELP nodejs_active_handles Number of active libuv handles grouped by handle type. Every handle type is C++ class name.
# TYPE nodejs_active_handles gauge
nodejs_active_handles{type="WriteStream"} 2 1574721817650
nodejs_active_handles{type="Server"} 1 1574721817650
nodejs_active_handles{type="Socket"} 1 1574721817650

# HELP nodejs_active_handles_total Total number of active handles.
# TYPE nodejs_active_handles_total gauge
nodejs_active_handles_total 4 1574721817650

# HELP nodejs_active_requests Number of active libuv requests grouped by request type. Every request type is C++ class name.
# TYPE nodejs_active_requests gauge
nodejs_active_requests{type="FSReqCallback"} 2

# HELP nodejs_active_requests_total Total number of active requests.
# TYPE nodejs_active_requests_total gauge
nodejs_active_requests_total 2 1574721817650

# HELP nodejs_heap_size_total_bytes Process heap size from node.js in bytes.
# TYPE nodejs_heap_size_total_bytes gauge
```

Upgrades

- Kubenetes – handles rolling out upgraded containers
- But not how to update Docker containers contents….
    - X projects * Y containers can be a big number
    - Which ones need to be upgraded?
    - Which versions of common components do they use?
- Imagine moving to a new team
    - What do their docker files look like
    - How are deployment artifacts organized
    - Endpoint names (live, ready, metrics or something else?)
    - Which logger
    - Etc….
- Consistency can help

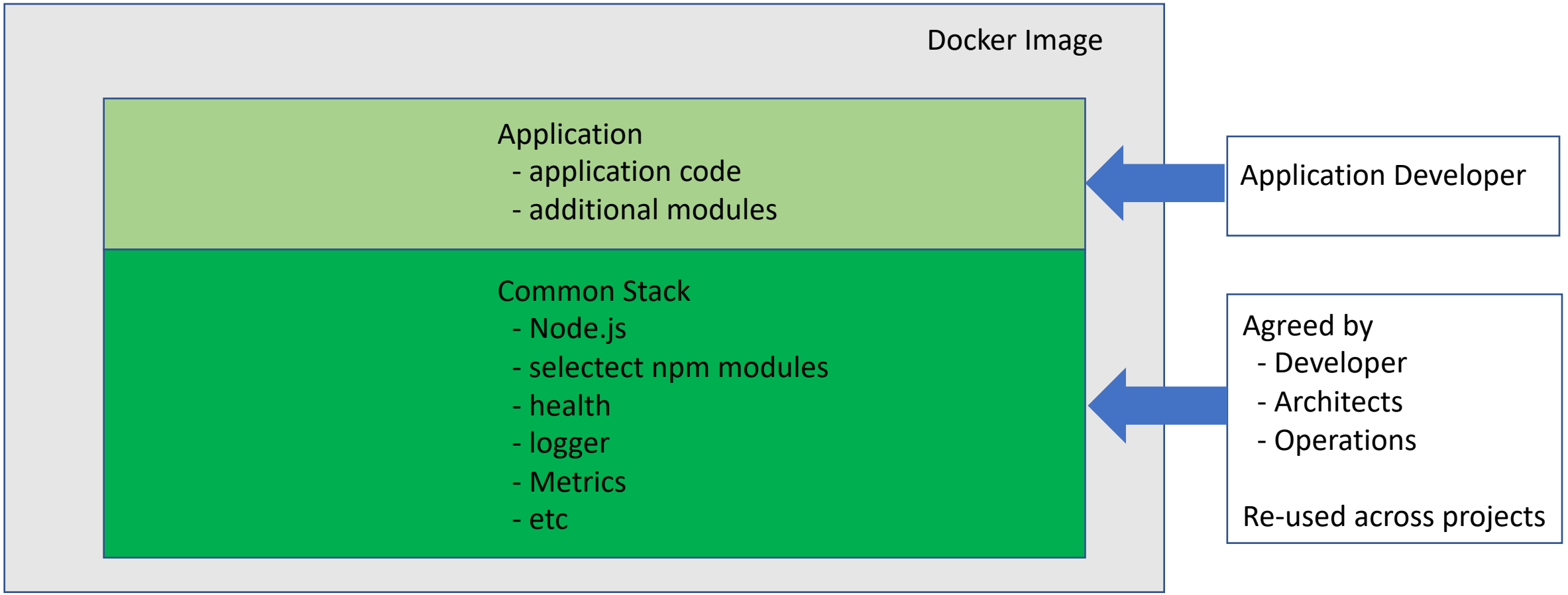
WHAT ?

WHAT ?

I just want to focus on the Application Code

Let’s Separate the Concerns

Tooling to build/test **efficiently** using best practices



Consistency…

Consistency…

…Not Uniformity

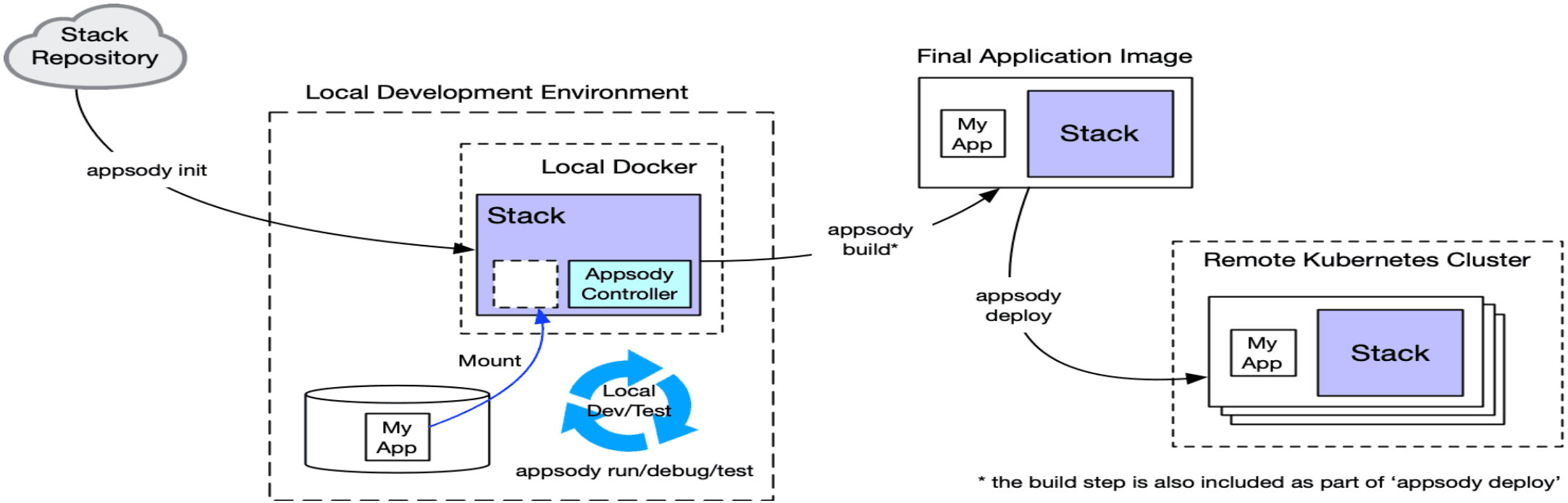
Tools ?

appsody

Appsody.dev

GitHub.com/appsody

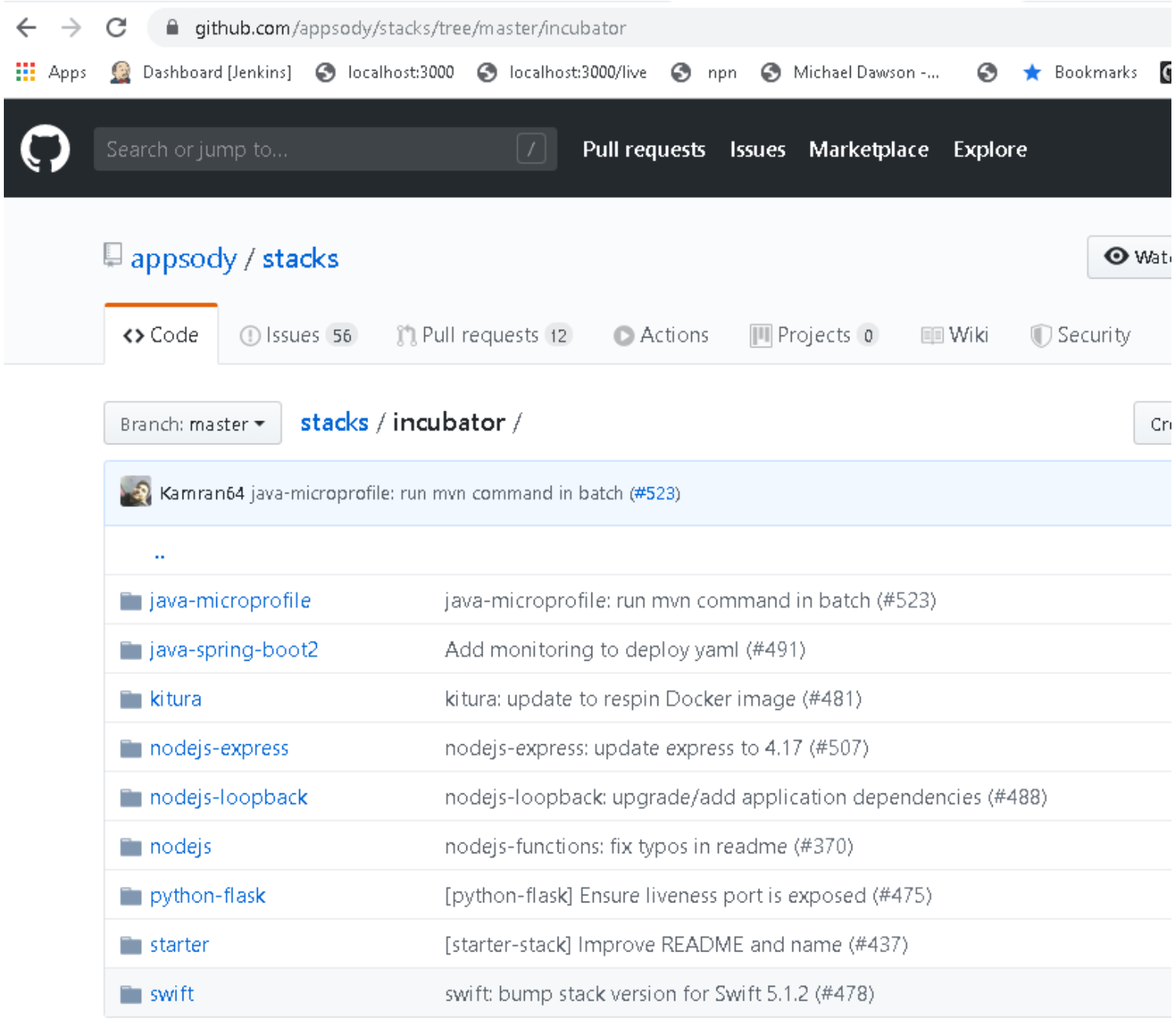
@appsodydev



Stacks

- Node.js Stacks
  - nodejs
  - nodejs-express
  - nodejs-loopback
  - nodejs-functions

FaaS meets Node.js Frameworks: Developing Cloud Native Node.js Applications at Speed !
https://sched.co/T5G6
11:40 immediately after this session.



Appsody – Demo

- Adds nice GUI
  - Appsody integration
  - Bring your own visual editor (ex VS Code)
  - Performance Monitoring
  - Tekton Pipelines
- https://developer.ibm.com/blogs/codewind-introduction/

Kabanero™

A modern microservices-based framework for continuous deploy of apps on Kubernetes and Knative

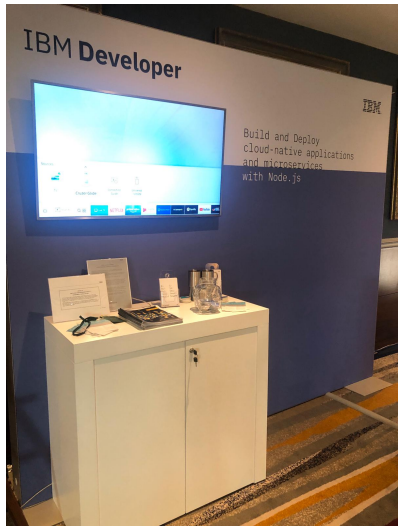
codewind
Dev Tools

appsody
Pre-built stacks

TEKTON
Pipelines

Built on popular open source projects

To learn more



- IBM Booth
  - Node.js Quick lab (15 mins)
  - Come talk to us

- Node.js On-line Lab (75 mins) - https://kabanero-workshop-nodejs.mybluemix.net/guides/dev-getting-started

- https://www.cloudnativejs.io/

Q/A

Copyright and Trademarks