How to do Continuous Delivery with Jenkins Pipeline, Docker and Kubernetes

James Strachan
How to do Continuous Delivery with Jenkins Pipeline, Docker and Kubernetes

James Strachan
Software is eating the world
Marc Andreessen - 2011
It is not necessary to change. Survival is not mandatory.
W. Edwards Deming
creating value through software is all about speed, iteration and Continuous Improvement
Continuous Assessment

Continuous assessment

Build

Release

Operator

Continuous integration and testing

Continuous delivery and deployment

Continuous operations

Test

Deploy

Monitor

Continuous assessment

#JenkinsWorld
how do we deliver software value fast?

Continuous Delivery of Containerised Microservices
microservices, microservices, microservices, ...

SAY MICROSERVICE

ONE MORE TIME
microservices are a way to develop software faster

• split monoliths into smaller microservices which release independently
• have small teams which look after the entire lifecycle of each microservice
Jenkins Pipeline rocks for Continuous Delivery!

• Isn’t my Jenkins server enough?
challenges of microservices

• lots of independent teams each want their own:
  • Jenkins server with their own pipelines
  • environments (Dev, Test, Staging, Production) to test/deploy in
  • resources (compute, network, storage)
• automation as lots of microservices
  • creation of projects, CD Pipelines, environments, releases etc!
we need a microservices platform!

- easy for independent teams to create, develop, manage microservices
- self service for speed
fabric8 microservices platform

https://fabric8.io/
fabric8 features

- **Create** wizards to create microservices
- **Build** packaging into immutable container images
- **Release** rolling upgrades across different environments
- **Runtime** service discovery, elastic scaling, failover, load balancing
- **Manage** centralise logs, metrics, alerts, tracing, circuit breakers
- **Feedback** dashboards and metrics to get feedback!
- **Platform** on premise, public or hybrid cloud
fabric8 stands on the shoulders of giants
fabric8 is microservices all the way down!
lets fabric8 microservices!!!!
get started with fabric8 on your laptop!

- download **gofabric8** binary:

- run this command:
  - **gofabric8 start**
get started with fabric8 on multi-node cluster

• create a kubernetes cluster:
  • Google cloud
    • https://cloud.google.com/container-engine/
  • Amazon, Azure, DigitalOcean etc:
    • https://stackpoint.io/
  • on premise:
    • http://kubernetes.io/docs/getting-started-guides/scratch/
• or create an OpenShift cluster:
  • https://www.openshift.com/
• then deploy fabric8 on the cluster:
  gofabric8 deploy
local Java development (pre-commit) with Maven


- setup your existing maven project for kubernetes:

- create a local cluster for development:
  mvn fabric8:cluster-start

- build and deploy your project into kubernetes:
  mvn fabric8:run
fabric8 stands on the shoulders of giants
architecture

- Jenkins for Pipelines
- Docker for packaging software as immutable images that are easy to run anywhere
- Kubernetes for orchestrating containers
  - keep containers running across a number of machines
  - deal with software, hardware and network failures
  - scales elastically
- Each team gets their own
  - Dev / Test / Staging / Production environments
    - each one is a separate kubernetes namespace
- Jenkins master
- Team dashboard, metrics, git repos etc
lessons learnt with Jenkins Pipelines for microservices...

• prefer docker images to ‘Jenkins tools’
  • its nicer to specify docker image + versions of tools inside Jenkinsfile!
    • works today with kubernetes-pipeline-plugin
    • coming soon in the new kubernetes plugin
  • mount secrets into build containers via kubernetes Secrets
  • copy the Jenkinsfile into each microservice git repo
• Jenkins master per team helps you go faster
  • teams can then configure their own Jenkins version, plugins and pipeline library
• use a library of Jenkinsfiles
  • most microservices of the same tech stack need very similar pipelines
  • then reuse common DSL / functions via plugins or shared library
• Jenkins Pipelines are awesome!!! :)}
## Development

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spring Boot</td>
<td>Spring based microservices</td>
</tr>
<tr>
<td>Spring Cloud</td>
<td>Kubernetes integration with Spring Cloud</td>
</tr>
<tr>
<td>Flux</td>
<td>Kubeflix</td>
</tr>
<tr>
<td>Function</td>
<td>Lambda style event based programming model for Kubernetes</td>
</tr>
</tbody>
</table>

## Continuous Improvement

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>Developer Console to help you create, build, manage microservices with deep visualisation into projects, apps and environments</td>
</tr>
<tr>
<td>Chaos Monkey</td>
<td>Randomly kills pods to help check your environment can withstand failures</td>
</tr>
<tr>
<td>Elasticsearch</td>
<td>Elasticsearch is a distributed fault tolerant scalable search engine for logs, events and documents</td>
</tr>
<tr>
<td>Prometheus</td>
<td>Service monitoring, metrics and alerting system with a time series database</td>
</tr>
<tr>
<td>Kibana</td>
<td>Let's view, filter and search all logs and events that have been written to Elasticsearch.</td>
</tr>
<tr>
<td>Grafana</td>
<td>Rich metrics dashboard and graph editor for Prometheus and InfluxDB</td>
</tr>
</tbody>
</table>

## Social

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slack</td>
<td>Public hosted chat service</td>
</tr>
<tr>
<td>Let's Chat</td>
<td>Open source on premise chat service</td>
</tr>
</tbody>
</table>
and more giants!...

### Systems Availability

<table>
<thead>
<tr>
<th><strong>Docker</strong></th>
<th>Linux container API, runtime, tooling and image registry</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Kubernetes</strong></td>
<td>Container orchestration</td>
</tr>
<tr>
<td><strong>OpenShift</strong></td>
<td>A PaaS (Platform As A Service) based on Docker and Kubernetes</td>
</tr>
<tr>
<td><strong>Kansible</strong></td>
<td>orchestrate operating system processes on Windows or Unix like Docker containers using Ansible</td>
</tr>
</tbody>
</table>

### Automation

<table>
<thead>
<tr>
<th><strong>Jenkins</strong></th>
<th>Continuous Integration and Continuous Delivery using reusable Jenkins Workflows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nexus</strong></td>
<td>Maven repository manager for Canary and Promoted releases along with a mirror of central maven repositories</td>
</tr>
<tr>
<td><strong>Hubot</strong></td>
<td>Chat bot with gateways to various chat services like Slack, Let's Chat, IRC and many others</td>
</tr>
<tr>
<td><strong>Arquillian</strong></td>
<td>System testing of Docker containers and Kubernetes resources in JUnit and Jenkins</td>
</tr>
<tr>
<td><strong>JBoss Forge</strong></td>
<td>A java toolbox to help create apps faster</td>
</tr>
</tbody>
</table>

### Transparency

<table>
<thead>
<tr>
<th><strong>Gerrit</strong></th>
<th>Code review of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Gogs</strong></td>
<td>On premise git repository hosting</td>
</tr>
<tr>
<td><strong>Taiga</strong></td>
<td>Project Management Tool (i.e. issue tracker and kanban board)</td>
</tr>
</tbody>
</table>
grafana dashboards for metrics - prometheus back end
ChatOps via hubot for IRC, Slack, LetsChat, HipChat etc
tracing with zipkin
circuit breaker with Hystrix

```java
@RequestMapping("/hello")
@HystrixCommand(fallbackMethod = "helloFallback", commandProperties = {
    @HystrixProperty(name = "execution.isolation.thread.timeoutInMilliseconds", value = "5000")
})
public String hello() {
    return restTemplateForObject("http://something/hello", String.class);
}

public String helloFallback()
    return "Hello fallback";
```

**Hystrix Stream:** http://turbine-server/turbine.stream

---

**Circuit**

<table>
<thead>
<tr>
<th>Quote</th>
<th>RequestQuoteCommand</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
</tbody>
</table>

- **Hosts:** 1090.995.995.995
  - Median: 1090.995.995.995
- **Requests:** 0.0/s
- **Cluster:** 0.0/s
- **Circuit Closed:**
  - Median: 1090.995.995.995

**Thread Pools**

<table>
<thead>
<tr>
<th>Bank</th>
<th>Bank2</th>
<th>BankController</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
</tbody>
</table>

- **Bank1**
  - Host: 0.0/s
  - Cluster: 0.0/s
- **Bank2**
  - Host: 0.0/s
  - Cluster: 0.0/s
- **BankController**
  - Host: 0.0/s
  - Cluster: 0.0/s

---

#JenkinsWorld
@RunWith(Arquillian.class)
public class KubernetesIntegrationKT {

    @ArquillianResource
    KubernetesClient client;

    @Test
    public void testAppProvisionsRunningPods() throws Exception {
        assertThat(client).deployments().pods().isPodReadyForPeriod();
    }
}


#JenkinsWorld
conclusions

- **Jenkins Pipelines** rock for Continuous Delivery
  - you all knew that right? :)
- **Kubernetes** is an ideal platform for CD, containers and microservices
  - [https://kubernetes.io/](https://kubernetes.io/)
- **Fabric8** an awesome open source microservices platform based on:
  - Jenkins, Docker, Kubernetes (and lots more great OSS)!

- **need help?** - join the fabric8 community:
  - or I’m [@jstrachan](https://twitter.com/jstrachan) on twitter - thanks for listening!!!