

cloud
bees®

CloudBees  DAYS



David Cañadillas
EMEA Solutions Architect and Technologist

"I love software tech and riding my Harley"

Agenda

Introduction

Kubernetes based apps

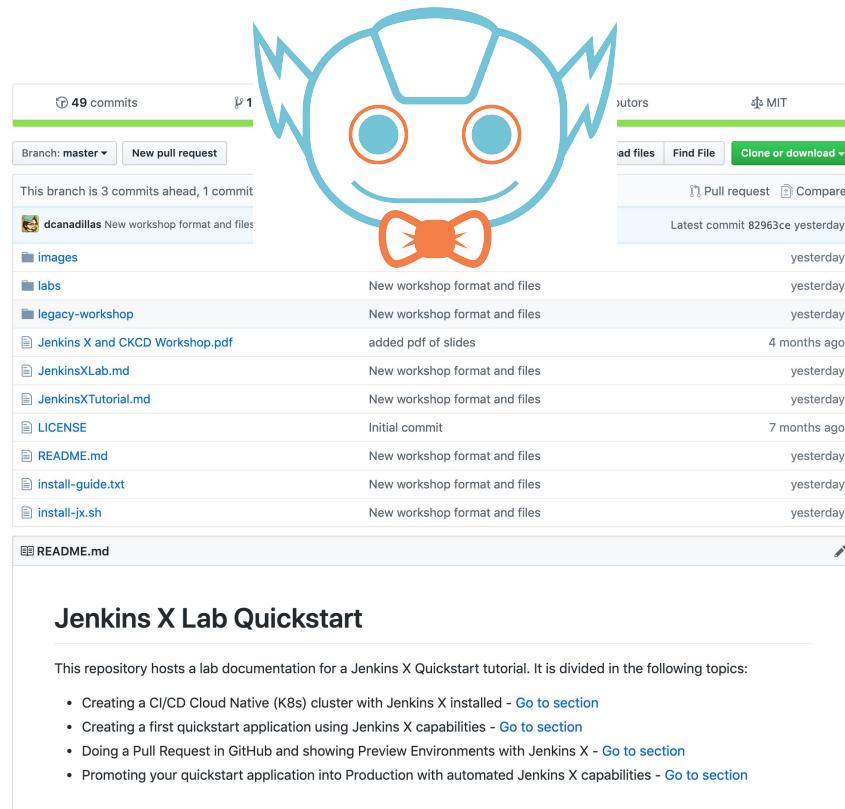
Jenkins X and Jenkins

GitOps

Library vs. Kubernetes Workloads

Build Packs

Jenkins X Workshop



A GitHub pull request interface for a repository titled "Jenkins X Lab Quickstart". The pull request has 49 commits and 1 review. The commit list shows various changes to files like README.md, JenkinsXLab.md, and LICENSE. A large, stylized cartoon robot head with orange eyes and a bow tie is overlaid on the right side of the interface.

File	Description	Time
Jenkins X and CKCD Workshop.pdf	added pdf of slides	4 months ago
JenkinsXLab.md	New workshop format and files	yesterday
JenkinsXTutorial.md	New workshop format and files	yesterday
LICENSE	Initial commit	7 months ago
README.md	New workshop format and files	yesterday
install-guide.txt	New workshop format and files	yesterday
install-jx.sh	New workshop format and files	yesterday
README.md		(Edit icon)

Jenkins X Lab Quickstart

This repository hosts a lab documentation for a Jenkins X Quickstart tutorial. It is divided in the following topics:

- Creating a CI/CD Cloud Native (K8s) cluster with Jenkins X installed - [Go to section](#)
- Creating a first quickstart application using Jenkins X capabilities - [Go to section](#)
- Doing a Pull Request in GitHub and showing Preview Environments with Jenkins X - [Go to section](#)
- Promoting your quickstart application into Production with automated Jenkins X capabilities - [Go to section](#)

Why should you want Native Kubernetes CD?

Scalability

K8s allows you to easily scale your CD workload up and down

Resilience

K8s is fault tolerant. You can't execute CD if your CD platform is down

Built-in objects

Resource, Config and Credential management is built-in. These objects form the core of any CD platform

Extensibility

K8s provides a number of extension points to include Custom Resource Definitions - ensuring that K8s is capable of providing a robust solution for any number of specialized use cases - like CD



How? - Best Practices for Native K8s CD

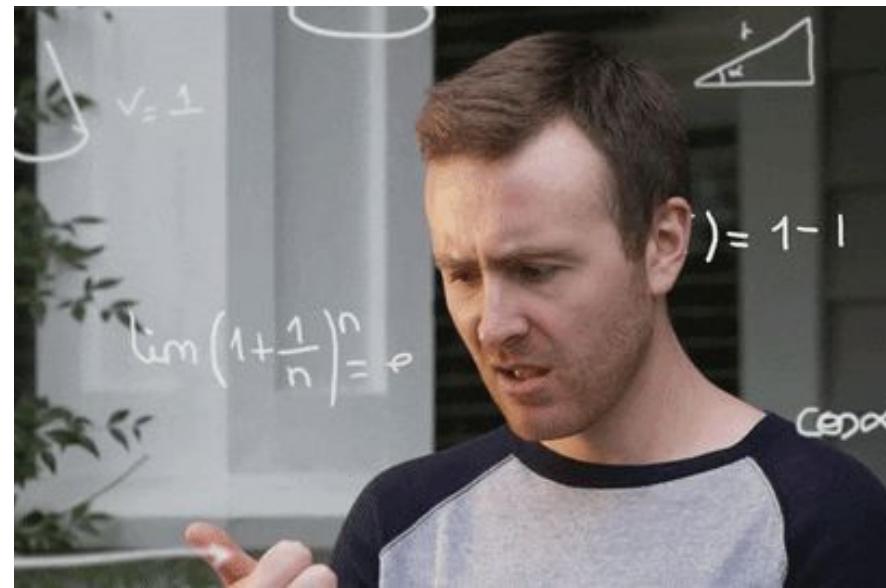
- Design cloud-native apps
- Adopt containers and schedulers
- Adopt Tekton as serverless solution for CI/CD pipelines in Kubernetes
- Adopt Prow (for GitHub¹) or Lighthouse for Git automation with GitOps and ChatOps
- Adopt Kaniko to securely build and push container images
- Adopt Helm as the standard Kubernetes packaging solution
- Join the tools and the processes into a single easy-to-use platform
- Create a CLI-first experience
- Define a prescriptive and easy to use process

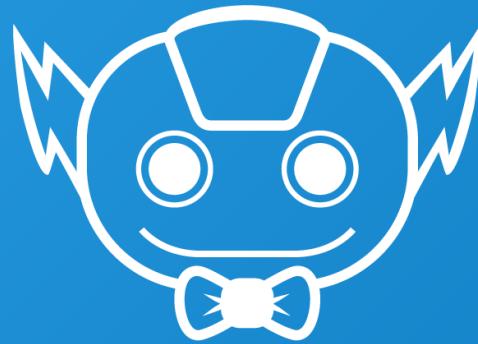


¹ Prow currently supports GitHub and GitHub Enterprise, there is an issue tracking support for other flavors of Git.

How do you do it?

- Are you an expert in kubernetes?
- Do you know how to create helm charts?
- Do you understand the intricacies of Prow and Tekton?
- How about Kaniko, Skaffold, Chartmuseum, Ksync, ...?
- Do you know how to configure GitOps and GitChat?
- Do you expect everyone in your organization to know all that?
- Do all your projects employ continuous delivery?
- If the answer to any of those is no, how do you plan to be competitive?





JENKINSX

<https://jenkins-x.io/>

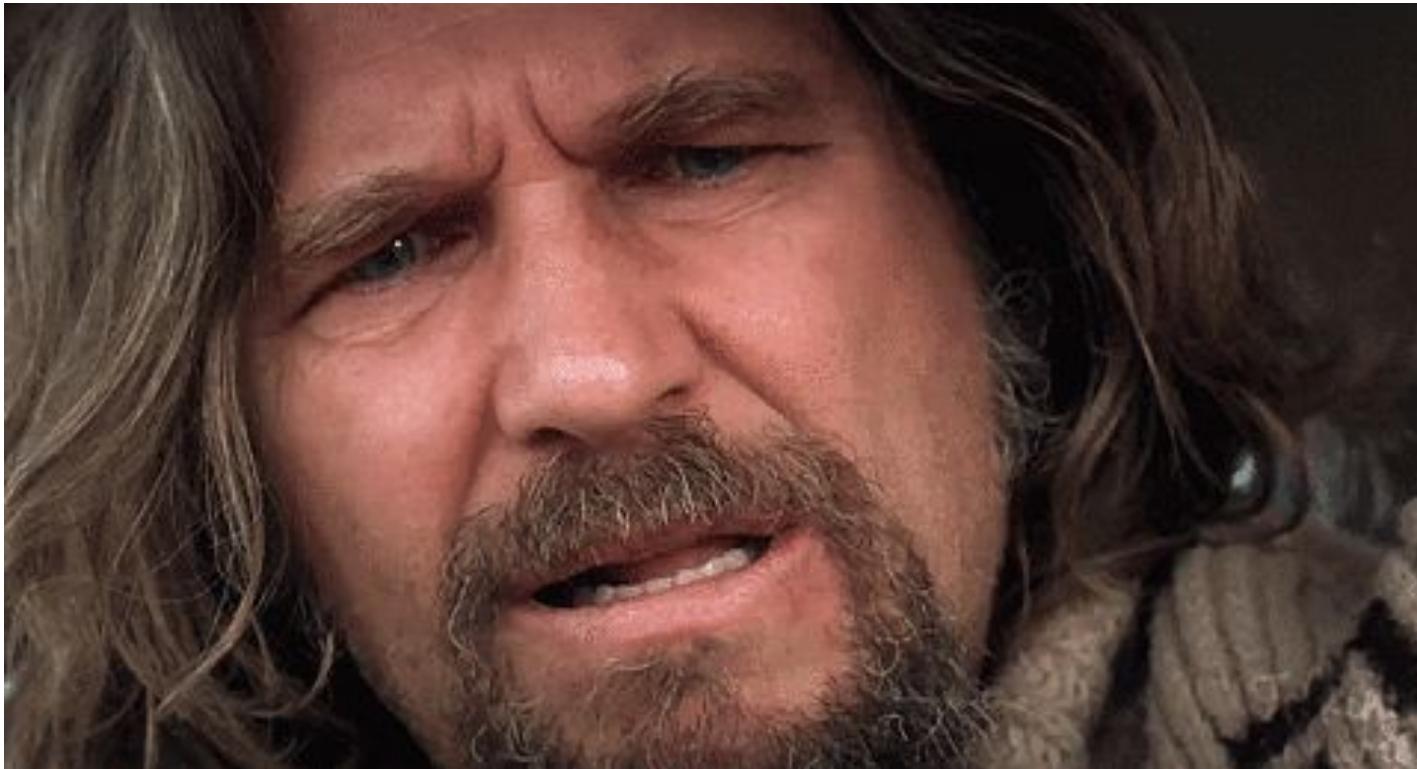
A Next Generation Cloud Native CI/CD

- Kubernetes native*
- GitOps promotions*
- Vault*
- Tekton CI/CD engine*
- Multi-cluster*
- Extensible pipelines*
- Build packs*



- ChatOps integration*
- Serverless*
- CloudBees Distribution*
- Preview Environments*
- Multi-Cloud*
- Easy on-boarding*
- Jenkins Apps*

You mean this is not even similar to Jenkins?



There are some differences



Must be installed on Kubernetes

vs

Installed Anywhere

Focused on K8s deployment

vs

Deploy Anywhere

CI/CD steps must run in containers

vs

CI/CD steps can run anywhere

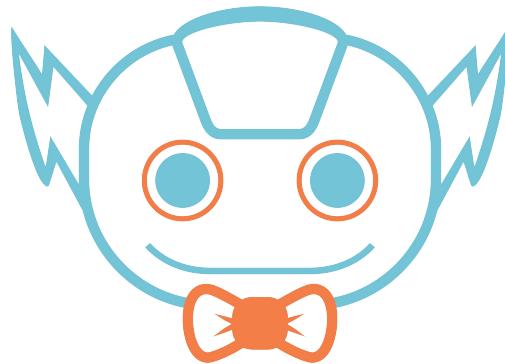
Prescriptive best practices

vs

Ultimate Flexibility



But they share some things



- CloudBees is the main contributor
- Leading CI/CD solutions
- Fast growing open source projects
- Can both run on Kubernetes



“One command to rule them all”

A CLI to start a real automation experience

```
$ jx create cluster gke --tekton --prow
```

```
$ jx get environments
```

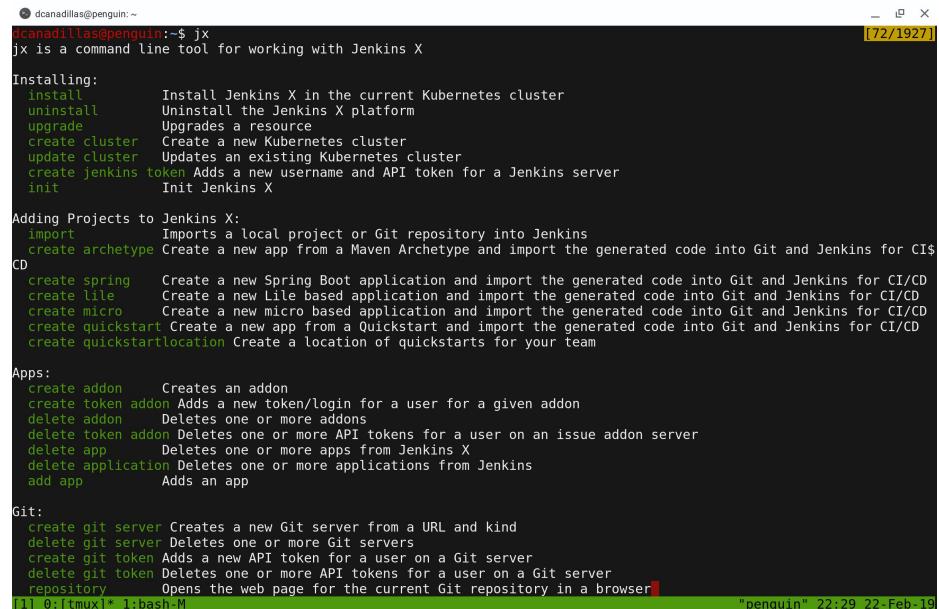
```
$ jx import --url  
https://github.com/dcanadillas/demo.git
```

```
$ jx get activity -f demo -w
```

```
$ jx get build logs  
<github-org>/demo/master
```

```
$ jx get applications
```

```
...
```



The screenshot shows a terminal window with the following content:

```
dcanadillas@penguin:~$ jx
jx is a command line tool for working with Jenkins X

Installing:
  install      Install Jenkins X in the current Kubernetes cluster
  uninstall    Uninstall the Jenkins X platform
  upgrade      Upgrades a resource
  create cluster Create a new Kubernetes cluster
  update cluster Updates an existing Kubernetes cluster
  create jenkins token Adds a new username and API token for a Jenkins server
  init         Init Jenkins X

Adding Projects to Jenkins X:
  import       Imports a local project or Git repository into Jenkins
  create archetype Create a new app from a Maven Archetype and import the generated code into Git and Jenkins for CI/CD
  CD
    create spring   Create a new Spring Boot application and import the generated code into Git and Jenkins for CI/CD
    create file     Create a new File based application and import the generated code into Git and Jenkins for CI/CD
    create micro    Create a new micro based application and import the generated code into Git and Jenkins for CI/CD
    create quickstart Create a new app from a Quickstart and import the generated code into Git and Jenkins for CI/CD
    create quickstart location Create a location of quickstarts for your team

Apps:
  create addon    Creates an addon
  create token addon Adds a new token/login for a user for a given addon
  delete addon    Deletes one or more addons
  delete token addon Deletes one or more API tokens for a user on an issue addon server
  delete app      Deletes one or more apps from Jenkins X
  delete application Deletes one or more applications from Jenkins
  add app        Adds an app

Git:
  create git server Creates a new Git server from a URL and kind
  delete git server Deletes one or more Git servers
  create git token Adds a new API token for a user on a Git server
  delete git token Deletes one or more API tokens for a user on a Git server
  repository      Opens the web page for the current Git repository in a browser
```

[1] 0:[tmux]* 1:bash-M "penguin" 22:29 22-Feb-19

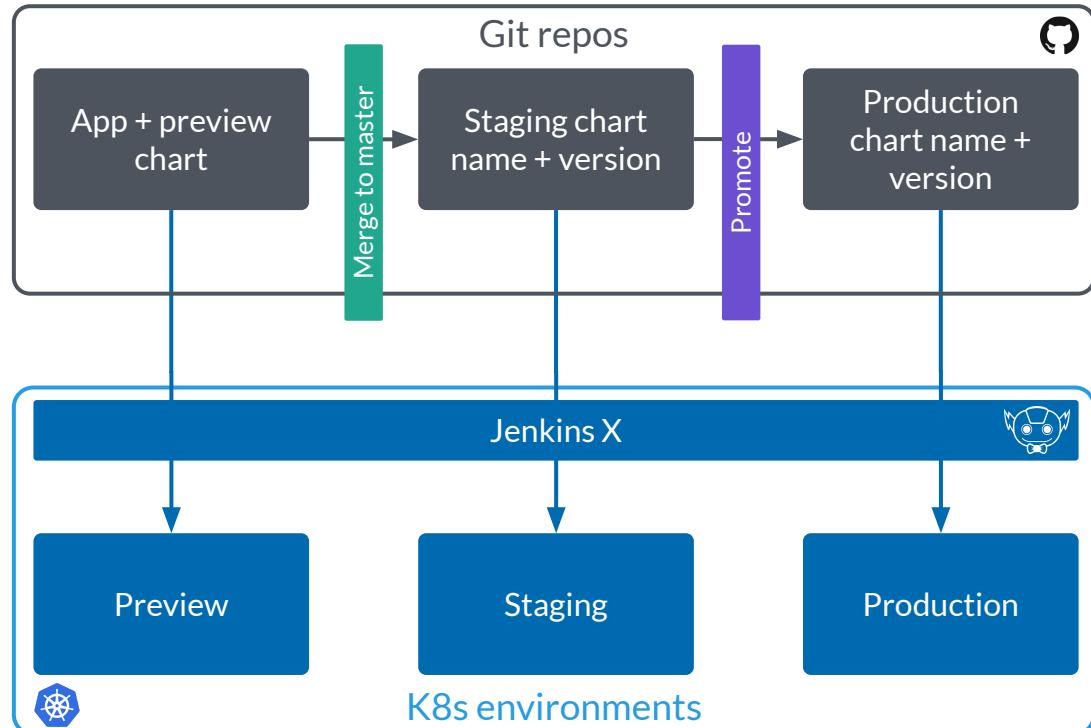


GitOps environments and promotion

Promotion automation

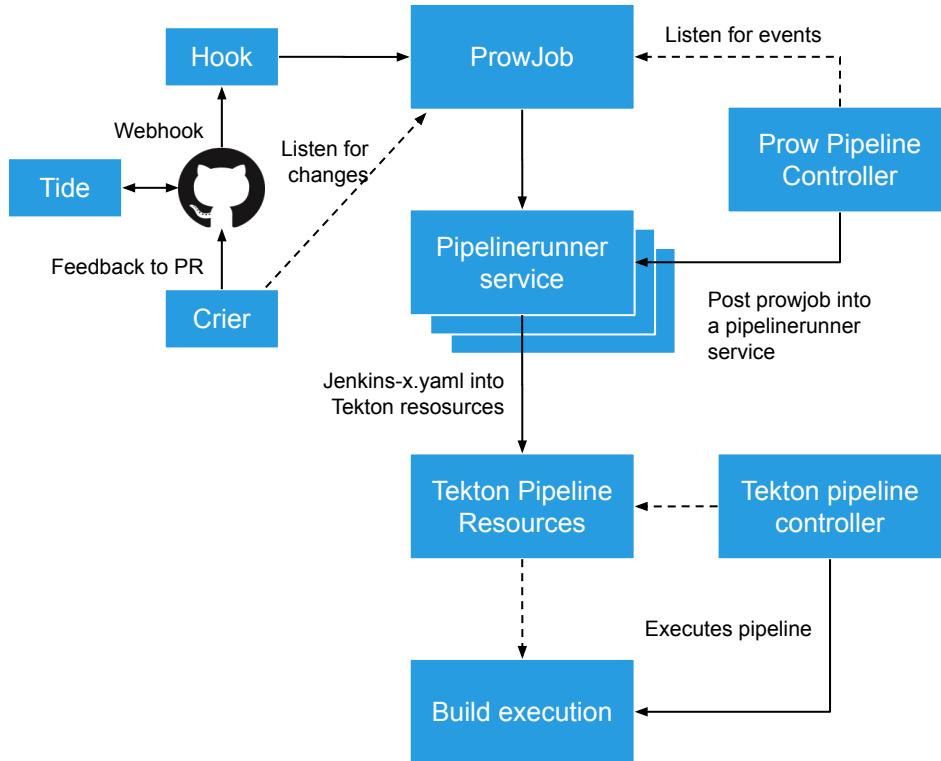
Thinking about true DevOps...
“Continuous Delivery meets Cloud Native”

- IaC
- Automated promotions
- Git as source of truth



Jenkins Serverless experience

Tekton based solution



Tekton pipelines

No Jenkins

Prow Webhook handler

Kubernetes Workloads vs Library Workloads

CI/CD or CI + Release



Library (classic) Workloads

- Jenkins X is not limited to applications targeting K8s for deployment
- Bring your own deployments and environments
- Library/Classic build pack supports CI+Releases but does not include CD.
 - e.g. do CI and release of your Java libraries or Node modules but don't deploy to Kubernetes
 - Extend the Library build pack with custom CD steps



Kubernetes Workloads

- This is what most people think of when they think of Jenkins X
- Includes deployment environments managed with GitOps with Staging and Production provided OOTB - modify or add additional environments
- Kubernetes Workload build packs support automated CI+CD with GitOps promotion and Preview Environments for kubernetes workloads



Build Packs

Developers to develop



How? - Best Practices for Native K8s CD

- Based on [Azure Draft Packs](#)
- Jenkins X provides two sets of build packs:
 - [jenkins-x-kubernetes](#)
 - [jenkins-x-classic](#)
- The K8s workload build packs extend the classic build packs
- Same build packs used for static and serverless
- Build packs are extensible



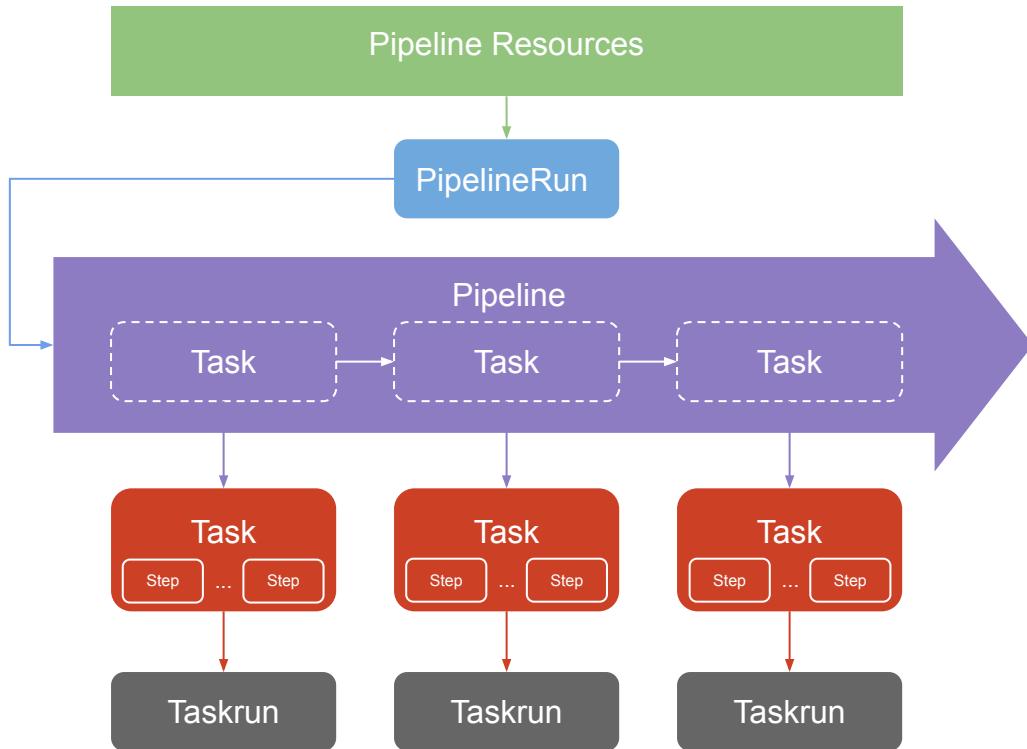
Custom build packs

- Fork <https://github.com/jenkins-x-buildpacks/jenkins-x-kubernetes.git>
- Add, modify, extend existing build packs in the fork
- Edit the build pack configuration for your team:

```
jx edit buildpack \
  -n kubernetes-workloads \
  -u https://github.com/{FORKED-ORG}/jenkins-x-kubernetes \
  -r master -b
```



What are Tekton Pipelines



Containers as building blocks

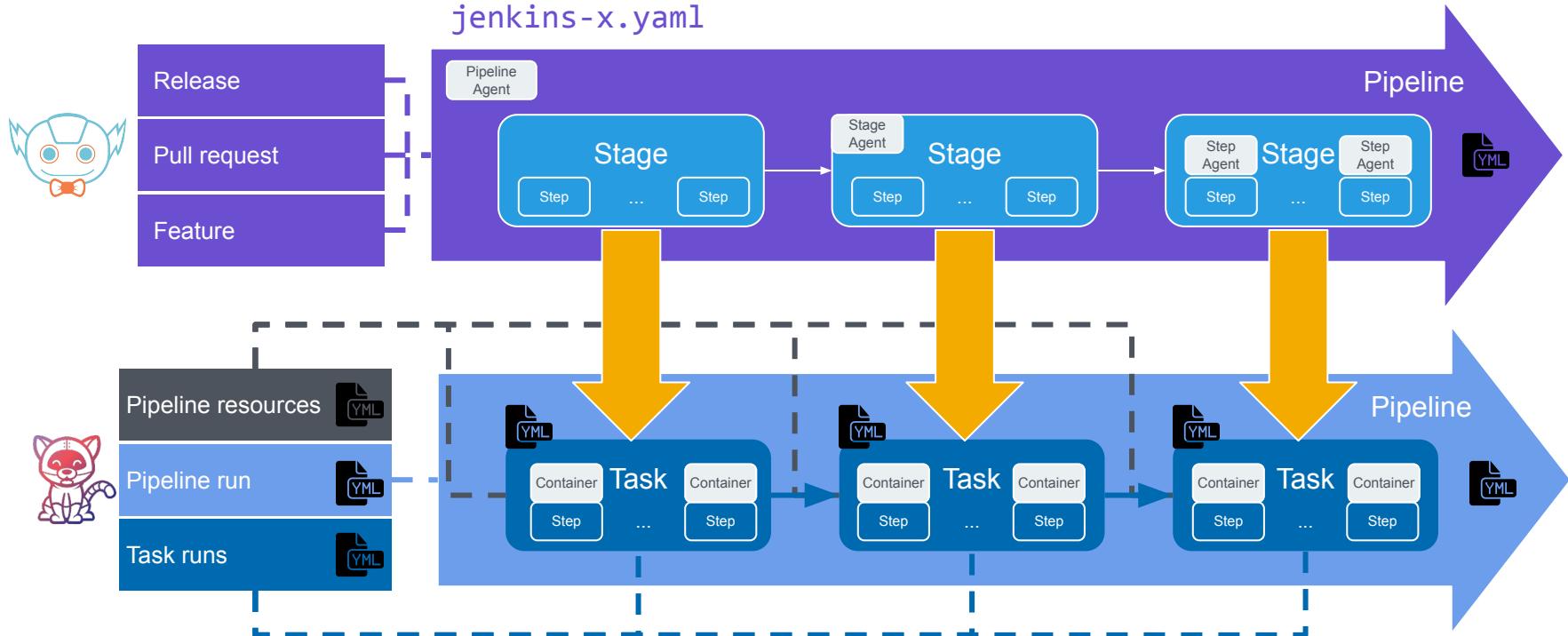
Deploy on any K8s cluster

Decouples stages into tasks

Decouples pipeline resources

Cloud Native CRDs

Decoupling Jenkins X Pipelines into Tekton



*Technologists post: <http://bit.ly/2kzusFz>

© 2019 CloudBees, Inc. All Rights Reserved.

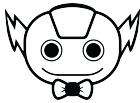
One more thing...

But is this production ready?



CloudBees Jenkins X Distribution

<https://www.cloudbees.com/products/cloudbees-jenkins-x-distribution>



Integrated Vault security

Certified distribution

Optional CloudBees support

Stable upgrades

GKE certified and more coming

Releasing a Jenkins X UI

The screenshot shows the CloudBees Jenkins X Distribution interface. At the top, there's a navigation bar with 'CloudBees Jenkins X Distribution', 'Projects', 'Builds', and 'Environments'. Below it, a sub-navigation bar shows 'Projects / dcandalilas / jx-go-Http' and 'master Build #5'. A message 'master--Build-#5 release' is displayed with a green circular icon. Below this, a section titled 'Update OWNERS' is shown.

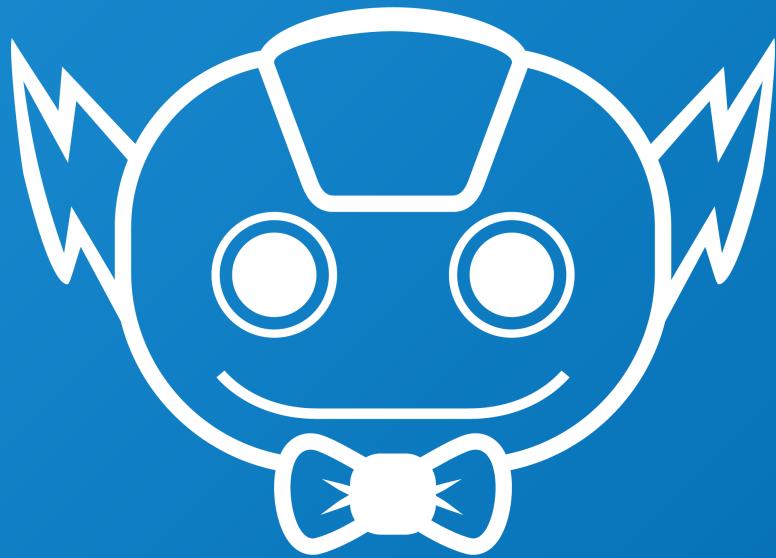
The main area displays a Jenkins pipeline build log. The log starts with 'Author: dcandalilas - Build started 4 hours ago' and continues with various log entries from 'INFO [WORKSPACE]'. The log includes commands like 'git clone https://github.com/dcandalilas/jx-go-Http.git --depth=1', 'cd jx-go-Http', 'mvn clean package', and 'java -jar target/jx-go-Http.jar'. It also shows 'Generated git credentials file /tmp/workspace/.jenkins/credentials' and 'Showing logs for build jx-go-Http#5'.

At the bottom of the screenshot, a terminal window titled 'david@dcandalilas-MBP jxui-backend-master' is open, displaying Jenkins X version information and system details:

```
david@dcandalilas-MBP jxui-backend-master % jx version
NAME          VERSION
jx             2.0.521
jenkins x platform 2.0.768
Kubernetes cluster v1.12.8-gke.10
kubectl        v1.14.1
helm client    Client: v2.13.1+g618447c
git            git version 2.21.0
Operating System Mac OS X 10.14.6 build 18G87
david@dcandalilas-MBP jxui-backend-master % cat ./jx/profile.yaml
installtype: cloudbees
david@dcandalilas-MBP jxui-backend-master %
```

The Workshop

Jenkins X Quickstart in GKE



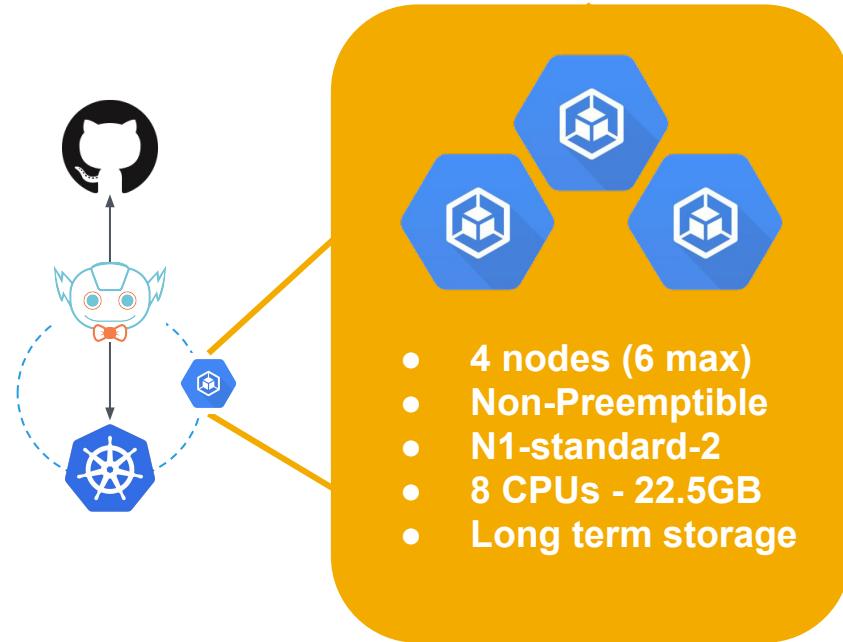
Start developing with Jenkins X

What you'll do

- Create a GKE cluster and install Jenkins X
- Use K8's GKE to deploy your application
- Create a Quickstart Project from a build pack
- Leverage Preview Environments for Pull Requests
- Promote your changes into Production

What you'll need

- GCP account (Free Tier can apply)
- Create a GitHub user and Org
- Internet access (Chrome recommended)



GitHub Workshop*: <https://github.com/dcanadillas/jenkins-x-workshop>

* The [CloudBees original repo](#) is outdated, and this one is an updated work in progress forked from the original



© 2019 CloudBees, Inc. All Rights Reserved.

The goal

This workshop goal is to **understand** Jenkins X components as well as the starting default workflow.

So:

- Follow every step of the lab
- Think about every cli command and type it
- Use “`jx <command> --help`”
- Work with your teammates
- Ask any question you have
- Share > Ask > Share



1 hour to complete
But... You can follow!



Do you have your own GCP account?

Be my guest!!



Forked repo *: <https://github.com/dcanadillas/jenkins-x-workshop>

* This GitHub repo is a work in progress than may be different from the CloudBees Workshop



© 2019 CloudBees, Inc. All Rights Reserved.



Thank You!

dcanadillas@cloudbees.com