WHITEPAPER

CI/CD for Cloud Native Applications on Kubernetes

Contents

INTRODUCTION

There are very few aspects that establish a common thread between diverse industries and organizations, one of which is the aspiration to deliver quality and secure products frequently and predictably to their customers. As a result, there is a critical need for a codified process model to represent a continuous pipeline that delivers code from version control to production in a deterministic fashion.

To address this gap, many products became available over the last decade in the continuous delivery market to respond to how software developers can build, configure, deploy, test and monitor their software. They solved bits and pieces of the larger puzzle, and expected the customer to assemble those pieces into a robust pipeline that satisfied their use cases. This assembly required deep and specialized knowledge of the continuous paradigm and customers had to make frequent trade offs between developing the product and getting entangled in shipping the product. The need for process-as-code, infrastructure-as-code and pipeline-as-code became even more crucial.

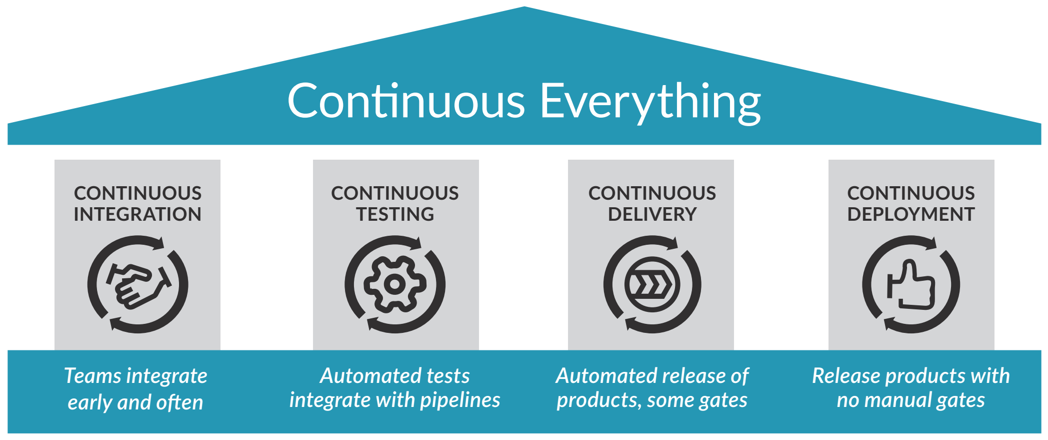
Around the same time, strong undercurrents emerged in other related industries like cloud, cloud native applications, microservices architecture and container orchestration. All of these technologies are instrumental in product releases, and these markets are now tied at their hips for success within the continuous pipeline model. Amazon Web Services, Microsoft Azure and Google Cloud emerged as leading cloud providers, and Kubernetes emerged as a leader in the container orchestration space. Kubernetes can be complex for organizations to setup and manage and so leading cloud providers started to offer managed Kubernetes solutions on their cloud to eliminate the complexity and lower the total cost of ownership.

In this heavily fragmented market, CloudBees stepped up to power the continuous economy and is building an end-to-end continuous delivery system. CloudBees is a major contributor of Jenkins X, an OSS CI/CD solution for modern cloud applications on Kubernetes. Jenkins X is a subproject of the Jenkins foundation, another OSS solution with more than 80% of total contributions coming from CloudBees.

This whitepaper goes into the details about the pillars of continuous everything and how Jenkins X automates these pillars in an opinionated fashion.

CONTINUOUS EVERYTHING

Continuous everything or the continuous paradigm encompasses many pillars. The ones relevant to this paper are:


Continuous Everything
CONTINUOUS INTEGRATION
CONTINUOUS TESTING
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT
Teams integrate early and often
Automated tests integrate with pipelines
Automated release of products, some gates
Release products with no manual gates

Source: CloudBees Inc.

Here are brief descriptions of each pillar so that readers can understand that continuous everything is a journey and not a destination. As they traverse through this paper, they will be able to appreciate the business value of having these pillars automated for them.

- **Continuous integration (CI):** Teams integrate early and often to detect issues earlier in the cycle. This directly translates to lower cost of execution.
- **Continuous testing:** Automated tests integrate with pipelines and factor into decisions involving promoting code and artifacts from one pipeline stage to the next.
- **Continuous delivery (CD):** Products are ready to be released from a source control repository to production in an automated fashion with one or more manual gates.
- **Continuous deployment (CD):** Release products from source control to production with no manual gates.

The risk of a defect during the software development and delivery process is:

- Small in the development and test phase
- Medium in the staging phase
- Large in the production phase

Fixing defects earlier in the pipeline leads to lower project costs.

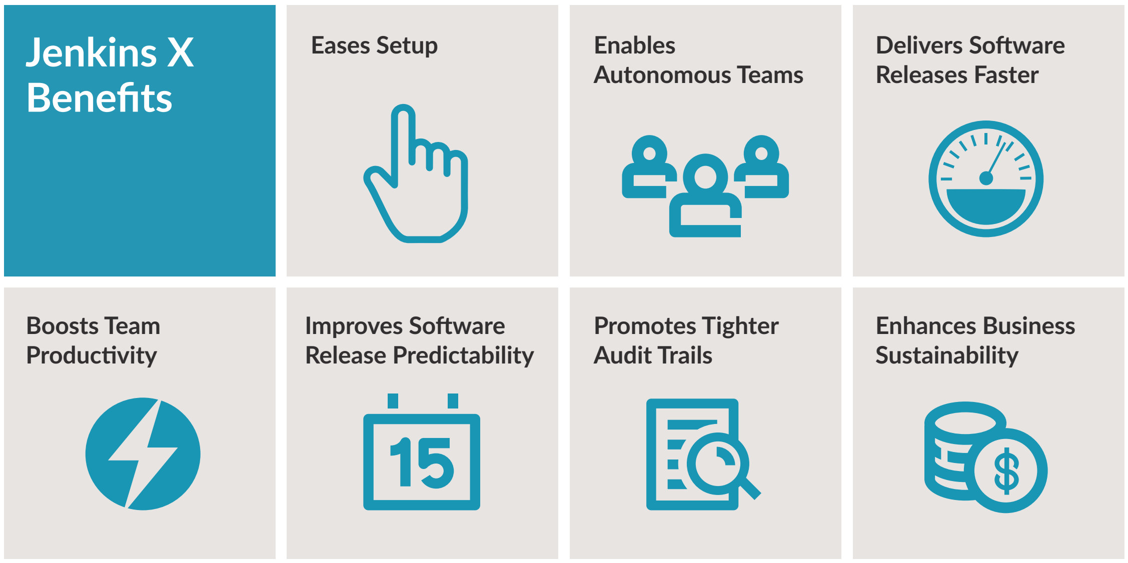
JENKINS X

Teams often spend days manually setting up Jenkins, pipelines, and implementing continuous integration/continuous delivery (CI/CD) effectively. To alleviate that problem, CloudBees, along with the Jenkins X community, developed Jenkins X, an automated CI/CD solution for modern cloud applications on Kubernetes.

Jenkins X comes with all the automation capabilities needed so you can focus on your application code. It automates your CI/CD needs on Kubernetes including:

- Automatic releases with semantic versions, creation of artifacts, Docker images, and helm charts
- Automatic promotion of versioned artifacts across your environments via GitOps
- Preview environments created on each pull request

Let's look at the following benefits that Jenkins X brings to organizations practicing continuous everything.


Jenkins X Benefits
Eases Setup
Enables Autonomous Teams
Delivers Software Releases Faster
Boosts Team Productivity
Improves Software Release Predictability
15
Promotes Tighter Audit Trails
Enhances Business Sustainability
$

Source: CloudBees, Inc.

Eases Setup

A SINGLE "JX" COMMAND TO RULE THEM ALL

Jenkins users appreciate the single Java command for installation. Jenkins X maintains this tradition and introduces the "*jx*" command line. Jenkins X has build packs for different kinds of projects, like Go, Node.js, Spring Boot and others to jump start users for their automation needs. It can recognize an existing project with "***jx import***", or create a new application with "***jx create quickstart***".

To create a new Kubernetes cluster on Google Cloud, use the command "***jx create cluster gke***" from the getting started guide. For those of you who have tinkered with Kubernetes, you will appreciate the managed Kubernetes services from the cloud providers. Jenkins X leverages these managed services and performs a lot of nitty-gritty details behind the scenes to simplify the workload of teams implementing continuous everything.

More importantly, you can clearly see the naval flavor on the Jenkins butler, who is fondly called "Captain Kubernetes"!

AUTOMATES INSTALL/CONFIG/UPGRADES OF TOOLS

Jenkins X automates the installation, configuration, and upgrades of tools like Helm, Skaffold, Jenkins, Monocular, Nexus, Docker, KSync, ChartMuseum, among others. This vastly simplifies the user experience, since users don't have to stomach all the various pieces on day one, and can learn at their own pace. With built-in automation capabilities, teams can focus on releasing quality and secure software frequently and predictably to customers, without worrying about the shipping logistics.

Enables Autonomous Teams

"*A chain is only as strong as its weakest link*" might be a cliché, but it holds true when it comes to defining organizations and teams. Think of the organization as the chain. If high-performing teams in the organization can move only as fast as the other teams, it might cause them to get frustrated and disengaged. Jenkins X enables teams to move at their own pace.

EACH TEAM GETS ITS OWN MASTER

Each team installs its own instance of Jenkins X, where they see their own pipelines in the user interface (UI). Each team can have its own master that won't interfere with other masters in the organization.

EACH TEAM GETS AN ELASTIC POOL OF PODS

Each team has their own Kubernetes build pods. Hence, teams can run at their own pace, and don't have to line up behind the entire organization's workload.

Increases Release Velocity

We often preach about the need for purposeful (not suicidal) speed without articulating exactly how that's done. Jenkins X takes an objective approach by creating assets you would otherwise have to manually create, which expedites development, testing and delivery.

CREATES YOUR GIT REPOSITORY

Whether you use "***jx import***" or "***jx create quickstart***", your code appears in a Git repository (e.g. GitHub). Git integration, along with webhooks to trigger pipelines on new commits, come out of the box. Teams can start to develop new features and tests right away, without getting bogged down by shipping logistics.

On each pull request, a CI pipeline is triggered to build your application and run all the tests. This keeps the master branch in a releasable state.

CREATES A KUBERNETES CLUSTER IN THE CLOUD

Jenkins X automatically creates a Kubernetes cluster in the cloud. For example, install Jenkins X on Google Cloud with a single "***jx create cluster gke***" command. Create a project on Google Cloud and use that project ID.

You could choose a different cloud of course, like Amazon Web Services (AWS) or Microsoft Azure, for instance. Provide the appropriate parameter to "***jx create cluster***" for that cloud provider as seen in the examples below:

» For Amazon Web Services, use "***jx create cluster eks***"

» For Microsoft Azure, use "***jx create cluster aks***"

» For Oracle Cloud, use "***jx create cluster oke***"

Boosts Team Productivity

Organizations attempt to measure productivity to enhance it. The following are classic ways to generate automated infrastructure-as-code. Jenkins X does this right out of the box.

CONTAINERIZES YOUR APPLICATION

Authoring immutable infrastructure can take some time, and time is money. Teams will appreciate that application containerization is automated in Jenkins X for applications that generate images. The Dockerfile is auto-generated.

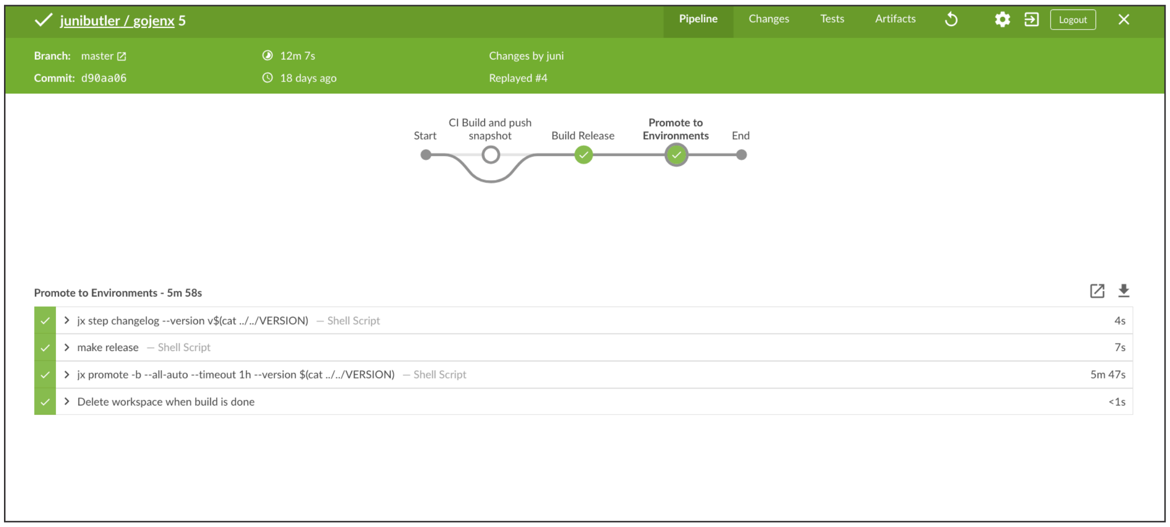
They say "*There ain't no such thing as a free lunch*". Well, in Jenkins X, not only do you get the lunch for free, you get the lunch box (the Docker container) as well.

PACKAGES AND DEPLOYS CONTAINERIZED APPS

Jenkins X makes releases boring. Helm charts appear in the automatically configured Helm repository. The application is packaged, deployed and run in a container in the Kubernetes cluster on the cloud of your choice.

DECLARES YOUR PIPELINE

Now guess the cost of generating your first pipeline. None! A neat "Jenkinsfile" is automatically generated by Jenkins X without you having to key in a single line of pipeline syntax. The default pipeline created by Jenkins X with three stages ("CI Build and push snapshot", "Build Release" and "Promote to Environments") is visualized below with Jenkins Blue Ocean.



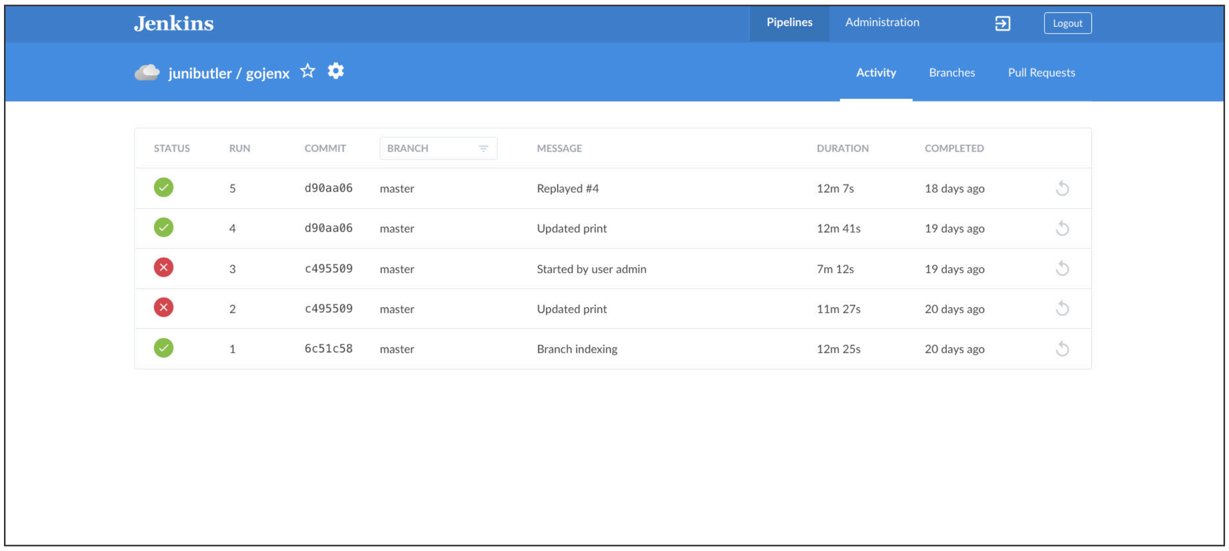
Promote to Environments - 5m 58s

```
✓ > jx step changelog --version v$(cat ../../VERSION)  — Shell Script                          4s
✓ > make release  — Shell Script                                                               7s
✓ > jx promote -b --all-auto --timeout 1h --version $(cat ../../VERSION)  — Shell Script   5m 47s
✓ > Delete workspace when build is done                                                       <1s
```

Source: CloudBees, Inc.

Users customize the various stages of the pipeline as needed. CloudBees recommends you practice declaring your pipelines rather than scripting them to prevent the “Jenkinsfile” from becoming a maintenance hazard.

From the command line, you can use '“***jx open jenkins***” or “***jx console***” to open the Jenkins UI in a browser. In the screenshot below, five runs of the pipeline have happened with numbers 1, 4 and 5 succeeding.



STATUS	RUN	COMMIT	BRANCH	MESSAGE	DURATION	COMPLETED
✓	5	d90aa06	master	Replayed #4	12m 7s	18 days ago
✓	4	d90aa06	master	Updated print	12m 41s	19 days ago
✗	3	c495509	master	Started by user admin	7m 12s	19 days ago
✗	2	c495509	master	Updated print	11m 27s	20 days ago
✓	1	6c51c58	master	Branch indexing	12m 25s	20 days ago

Source: CloudBees, Inc.

Improves Predictability of Releases

Another term thrown around in the continuous everything world is “predictability”. Vendors market pipelines as tools that make releases predictable, without understanding how it happens. Jenkins X does exponentially better than its competition by not just providing the following features, but also by automating these with simple command line options.

PRE-COMMIT VALIDATION IN PRODUCTION

Engineers test on local systems before committing code but local validation on laptops can be error-prone due to missing and/or incorrect dependencies.

Using "***jx create devpod***", you could create a dev/test environment using the exact build pods that the pipeline will use in production. With "***jx sync***", you could sync your source code inside a container in the Kubernetes cluster on the cloud. Hence, "works in my environment" is a thing of the past.

VALIDATION IN A PREVIEW ENVIRONMENT

The preview environment is equivalent to a pre-production or an integration environment, where you can validate your application's behavior alongside other apps, if you choose to. Here's where you see the world in the same light as your customer would, and that raises confidence before you promote your code to production.

Promotes Auditability

Git is the single source of truth in the Jenkins X world. Audit trails are available in Git, thanks to GitOps being the norm. Everything is versioned, and hence production outages can be debugged with ease.

GITOPS

Sharing is not always caring - teams step on each others' toes when they share static environments. GitOps enables teams to define their environments in an inexpensive and scalable fashion. Jenkins X uses GitOps to manage configurations for environments and applications, and also application versions.

Two environments – staging and production – are created by default as Git repos, and they do not (and should not) share resources. Code promotion happens:

» Automatically to staging

» Manually to production using the "***jx promote...***" command

The number of environments and their names are configurable with the "***jx create environment***" and "***jx edit environment***", although we recommend you take a minimalist approach. As far as environments are concerned, the less the merrier, since each additional environment adds to maintenance overhead.

The presence of one (or more) manual gate(s) is in fact the primary difference between continuous delivery and continuous deployment. Some organizations do not allow headless end-to-end runs all the way to production. To remove all manual gates, you may have to wade through both technical challenges and political power plays. Jenkins X makes these configurable and you can remove the manual gate(s) when you have built in the organizational discipline.

Enhances Business Sustainability

Successful businesses not only just deliver the goods, but they can do so consistently for years and decades. These businesses run as marathons, not sprints, since sustainability is a key part of their strategy. Jenkins X takes automation to the next level and prevents burn-out among software developers, quality assurance, release engineers and others who would otherwise have to do this manually. And, it introduces techniques to experiment with new ideas without inflating the budget. Here's how.

EASES ROLLBACKS

Jenkins X reduces shelf time of new features and improves "Time-to-Market" without burning out team members. Teams can:

- Experiment with new ideas safely
- Pivot quickly if the experiment fails.

Although continuous delivery advocates for roll forward only, the capability to rollback is essential, although you may choose to not use it. In Jenkins X and GitOps, application versions are Git tags. Releases are done from the master branch, and you can see the release versions under the Release Tab in GitHub. If you choose to, you can revert to the last known good version easily with the "***jx promote…***" command.

LOWERS ADOPTION COST

There is one important question that should be answered by teams doing tool evaluations. If the proof-of-concept of a new tool fits your bill, what would be the transition cost? That is, how many person hours would it take to migrate away from your current approach? Many successful proof-of-concepts never see the light of day simply because the migration cost is prohibitively expensive and interferes with the team's "keep the lights on" responsibilities.

Jenkins commands the lion's share of the market in the world of pipelines. According to community stats, Jenkins sites report their usage to the Jenkins project which runs over 19 million jobs every month. Every small, medium and large company banks on the fact that most or almost all developer new hires come "equipped" with Jenkins knowledge. That radically reduces a key business metric "Time-to-Code" – which is the time taken for a new hire to build, configure, deploy, test and release their first lines of code to production.

The powerhouse beneath Jenkins X is Jenkins. So, if you are familiar with Jenkins pipeline-as-code, you are in for a treat. Additionally, Jenkins X automates the installation, configuration and upgrade of Jenkins and other open source tools in a way so teams can focus on building products that wow their customers.

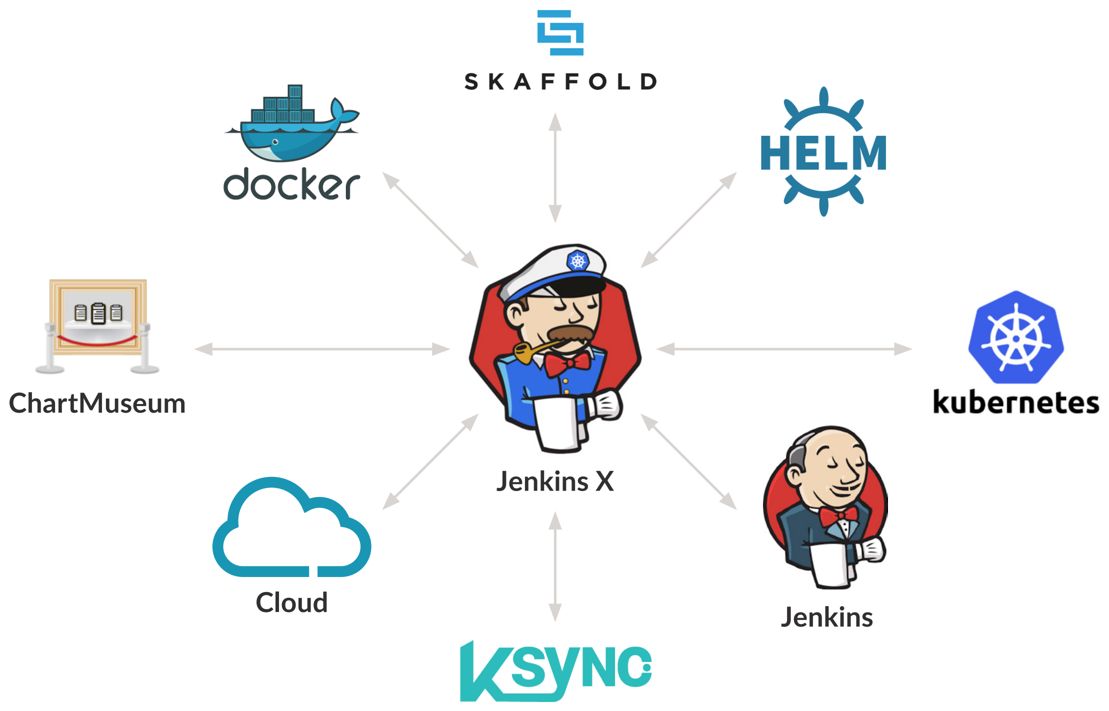
The plethora of technologies and concepts and how they interact can be intimidating in the beginning...

- Continuous everything (integration | testing | delivery | deployment | analytics)
- GitOps
- Git and version control
- Containers and Docker
- Skaffold, which facilitates CD for Kubernetes applications
- Container orchestration and Kubernetes
- Helm charts, that define, install, and upgrade apps in Kubernetes
- ChartMuseum, an open-source Helm chart repository
- Managing Kubernetes on a cloud of your choice
    - Google Cloud
    - Amazon Web Services
    - Microsoft Azure
- Jenkins declarative pipeline syntax and “Jenkinsfile”

...and hence the automated CI/CD solution helps you focus on writing quality and secure applications instead of burning the midnight oil on infrastructure.

Under the Hood

Jenkins X gets you started at the speed of light, however, the onus lies on you to gradually and eventually learn these underlying details. While we are sure you are well on your way to design and implement resilient pipelines, we wanted to make sure you have these details at your fingertips. Here is a quick overview on some of those "undercover agents", so that you can seamlessly continue on your journey in the continuous everything paradigm.



Source: CloudBees, Inc.

JX

Jenkins X introduces the "*jx*" command line - a single command to rule them all. You can either:

- Install the "*jx*" binary on your local machine, or
- Use your favorite cloud's shell

For example, the Google Cloud shell comes with most of the tools you will need like Git, gcloud, kubectl etc, and is the recommended option.

Some examples of using “*jx*” are:

- “***jx create cluster gke***” creates a new Kubernetes cluster on Google Cloud
- “***jx create quickstart***” creates a new app and imports the generated code into Git and Jenkins for CI/CD
- “***jx open jenkins***” opens Jenkins in a browser
- “***jx sync***” synchronises your local files to a devpod

You can type “***jx help***” to find the various operations you can perform in Jenkins X.

HELM

Helm charts manage Kubernetes applications in a predictable and repeatable fashion. This enables them to:

- Define the app
- Install the app
- Upgrade the app

The Helm charts are versioned in source control, and can be audited. They are also easy to create and publish and reduce communication overhead. Finally, Helm charts make rollbacks to the last known good version stress-free (and drama-free).

CHARTMUSEUM

You need to understand what a chart repository is first before learning about ChartMuseum.

A chart repository is an HTTP server that houses an “index.yaml” file and optionally some packaged charts. The repository is also a location where packaged charts can be stored and shared.

ChartMuseum is an open-source Helm chart repository written in Go. It is compatible with cloud storage backends like:

- Google Cloud Storage
- Amazon S3
- Microsoft Azure Blob Storage
- Alibaba Cloud OSS Storage
- Openstack Object Storage

ChartMuseum can:

- Generate the repository index ("index.yaml") based on packages found in storage:
    - You are no longer required to maintain your own version of "index.yaml"
    - If you store your own version of "index.yaml", it will be completely ignored
- Provide a CLI for:
    - Installation
    - Configuration
- Provide an API for:
    - Uploading a new chart version
    - Deleting a chart version
    - Uploading a new chart package to storage
    - Listing all the charts

SKAFFOLD

Skaffold is a command line tool that facilitates CD for Kubernetes applications. It can iterate on your application source code locally then deploy to local or remote Kubernetes clusters. Skaffold also handles the workflow for building and deploying your application, and leverages that workflow in an automated pipeline to promote applications from development and testing phases to staging to production.

JENKINS

Jenkins is an open source automation tool that is a self-contained Java-based program used as a simple CI server or turned into a continuous delivery hub. The tool has a plug-in ecosystem to support the building, testing and deployment of any project.

The tool supports a "Jenkinsfile" that leverages pipeline-as-code syntax or DSL (domain-specific language) in two formats:

- Declarative syntax (recommended)
- Scripted syntax

DOCKER

Docker is a software platform for container application development. It helps developers:

- Standardize and automate the way we build, manage, and secure applications using containers
- Create true independence between applications and infrastructure
- Create a model for better collaboration and innovation with QA, release, operations, security and other related functions

KUBERNETES

Kubernetes runs production workloads at Google, which translates to running billions of containers per week. Among many other things, Kubernetes:

- Automates deployment, scaling and management of containerized apps
- Groups containers that make up an application into logical units for easy management and discovery
- Gives you freedom to take advantage of on-premise, hybrid or public cloud infrastructure, letting you effortlessly migrate workloads
- Restarts containers that fail
- Replaces and reschedules containers when nodes die
- Kills containers that don't respond to health checks
- Scales your application up and down:
    - Automatically based on CPU usage
    - Manually with a command or a UI

KSYNC

Ksync syncs files between our local file system and a Kubernetes cluster. The tool transparently updates containers running on the cluster from your local checkout. Ksync also enables developers to use their favorite IDEs (integrated development environment) to work from inside a cluster instead of outside it.

"***jx sync***" syncs files between our local file system to the devpods on the Kubernetes cluster. This prevents typical problems like "works on my machine" from slowing down business, and lets everyone channel their energy into supporting the customer.

CLOUD

Kubernetes is difficult to setup and manage. Jenkins X alleviates that pain and leverages the managed Kubernetes solutions offered by major cloud providers, like:

» Google Kubernetes Engine (GKE)

» Amazon Elastic Container Service For Kubernetes (EKS)

» Microsoft Azure Kubernetes Service (AKS)

» Oracle Kubernetes Engine (OKE)

Let's take the example of Google cloud.

The Google Cloud SDK provides the command line interface for Google Cloud Platform products and services, like:

» Gcloud, which manages authentications, local configurations, developer workflow and interactions

» Kubectl, which orchestrates the deployment and management of Kubernetes container clusters on gcloud

Pipeline is No Longer a Pipe Dream

Jenkins X automates CI/CD for cloud applications on Kubernetes, so that engineers can focus on writing and testing products. Jenkins X abstracts a lot of the shipping logistics that would otherwise consume your time and energy.

The Jenkins X community is growing rapidly and the project is moving fast. As we add new capabilities and techniques to Jenkins X, we'll make sure to let you know through the CloudBees blog and on Twitter @cloudbees. You can also stay current on Jenkins X news or follow @jenkinsxio on Twitter too.

Try out Jenkins X and help shape its future. Let us know what issues you face. Pipeline is no longer a pipe dream – so, happy coding.

CloudBees, Inc.
2001 Gateway Place, Suite 670W | San Jose, CA 95110 | United States
www.cloudbees.com | info@cloudbees.com