



# The Difference between VM & Docker-based Pipelines

GUY SALTON

# Guy Salton

*Solution Architect*



*[guy.salton@codefresh.io](mailto:guy.salton@codefresh.io)*



# Agenda

<https://github.com/containers101/docker-based-pipelines-webinar/>

1. Docker usage in Continuous Integration
  2. Pre-install vs Dockerizing build tools as pipeline steps
  3. Upgrading build tools to new versions
  4. Mixing multiple versions of the same tool in the same pipeline
  5. Creating new pipeline steps on the fly
- **Demos for everything using Codefresh and Jenkins**

# Theory: Docker-based Pipelines

# “Docker-based” means 2 different things:



**Using Docker as a  
deployment package**

(this is what most people think)



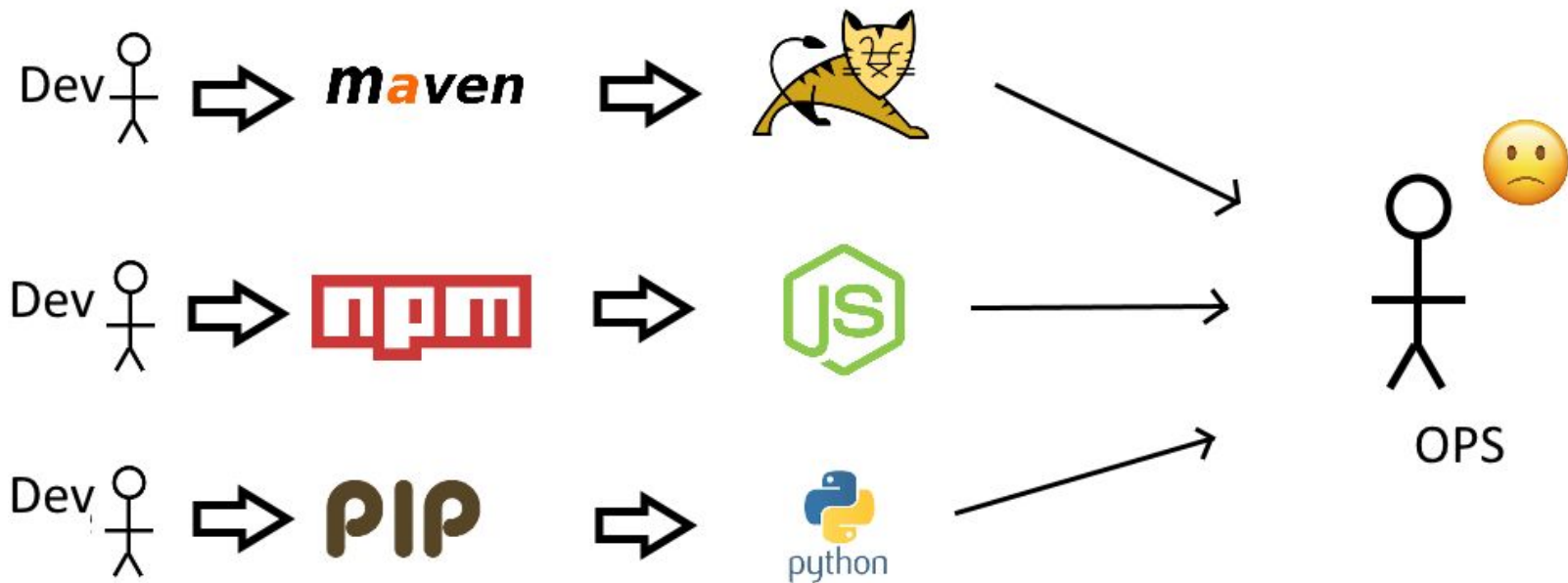
*90% of cases: “We have migrated  
to Docker in production”*



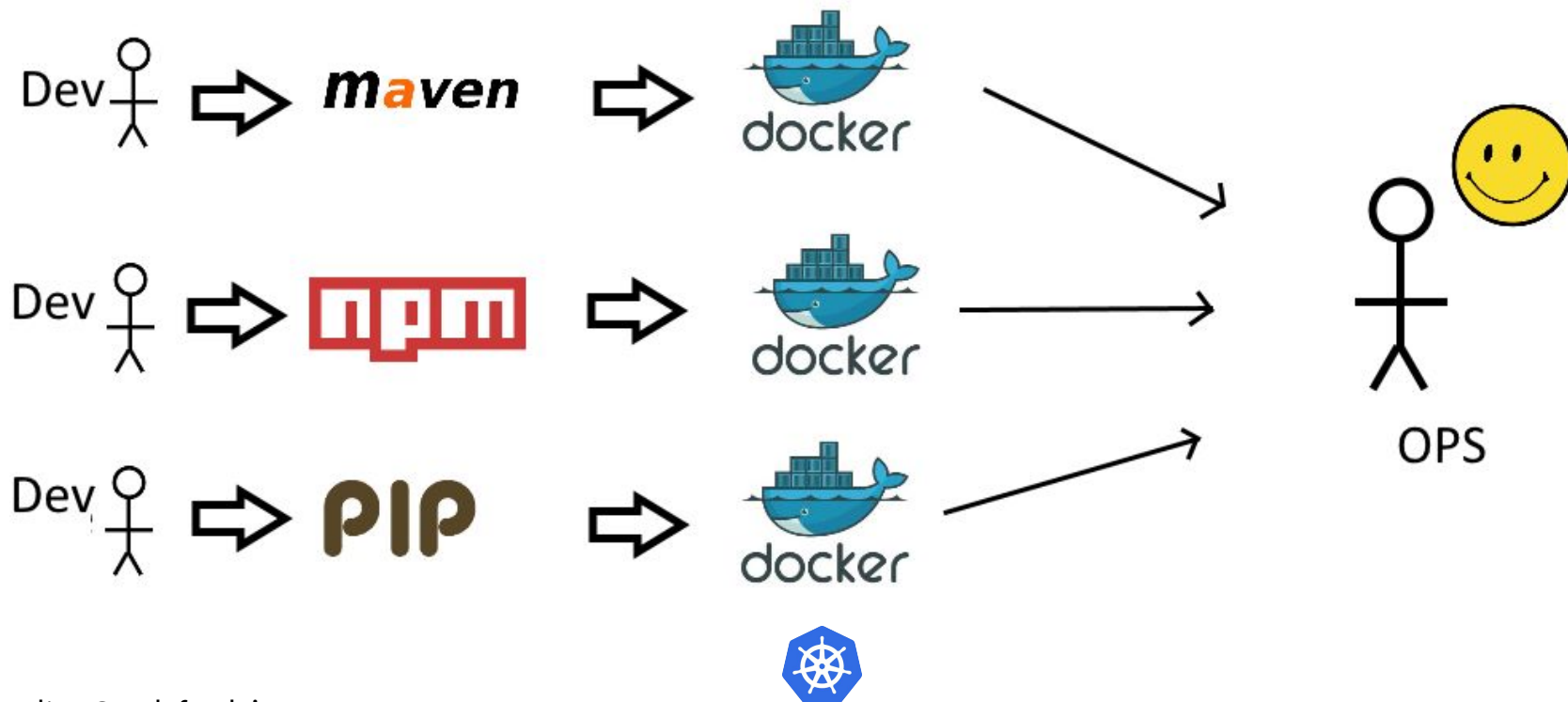
**Using Docker for build Tooling**

(this is not what most people think)

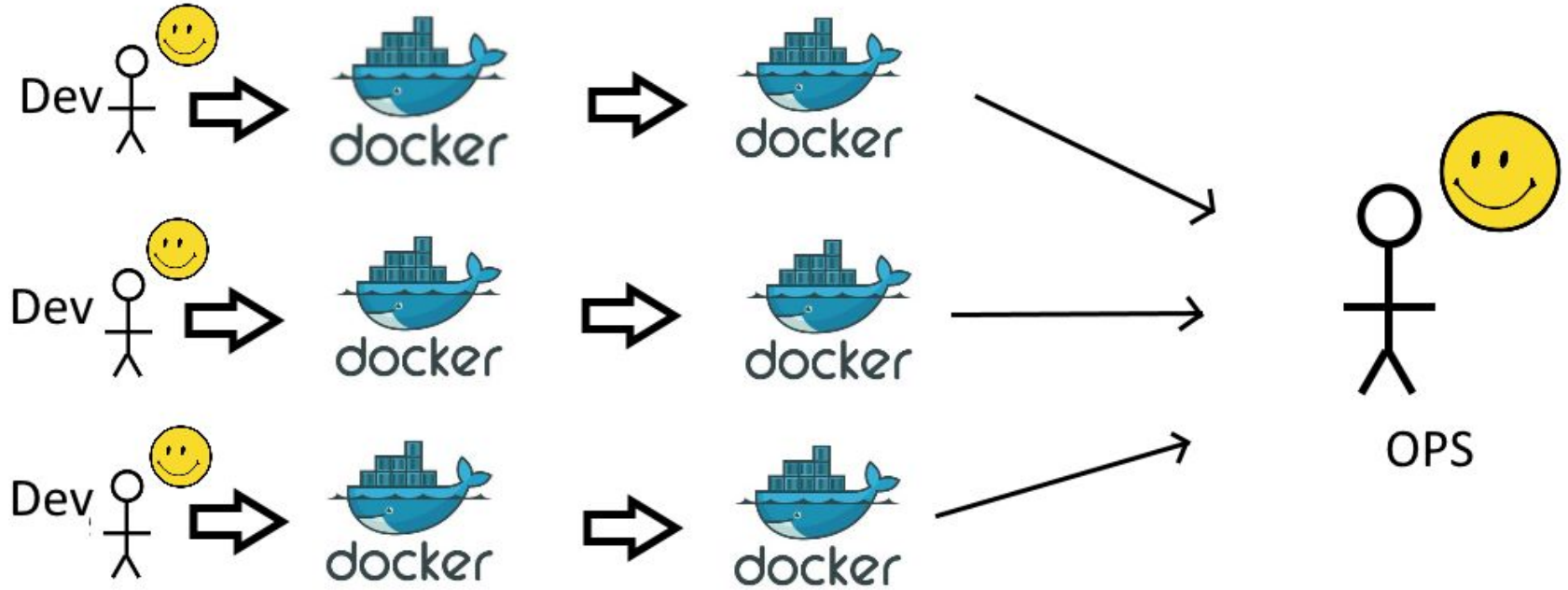
# Before Docker – The Dark Ages



# Docker-based Deployments - Better

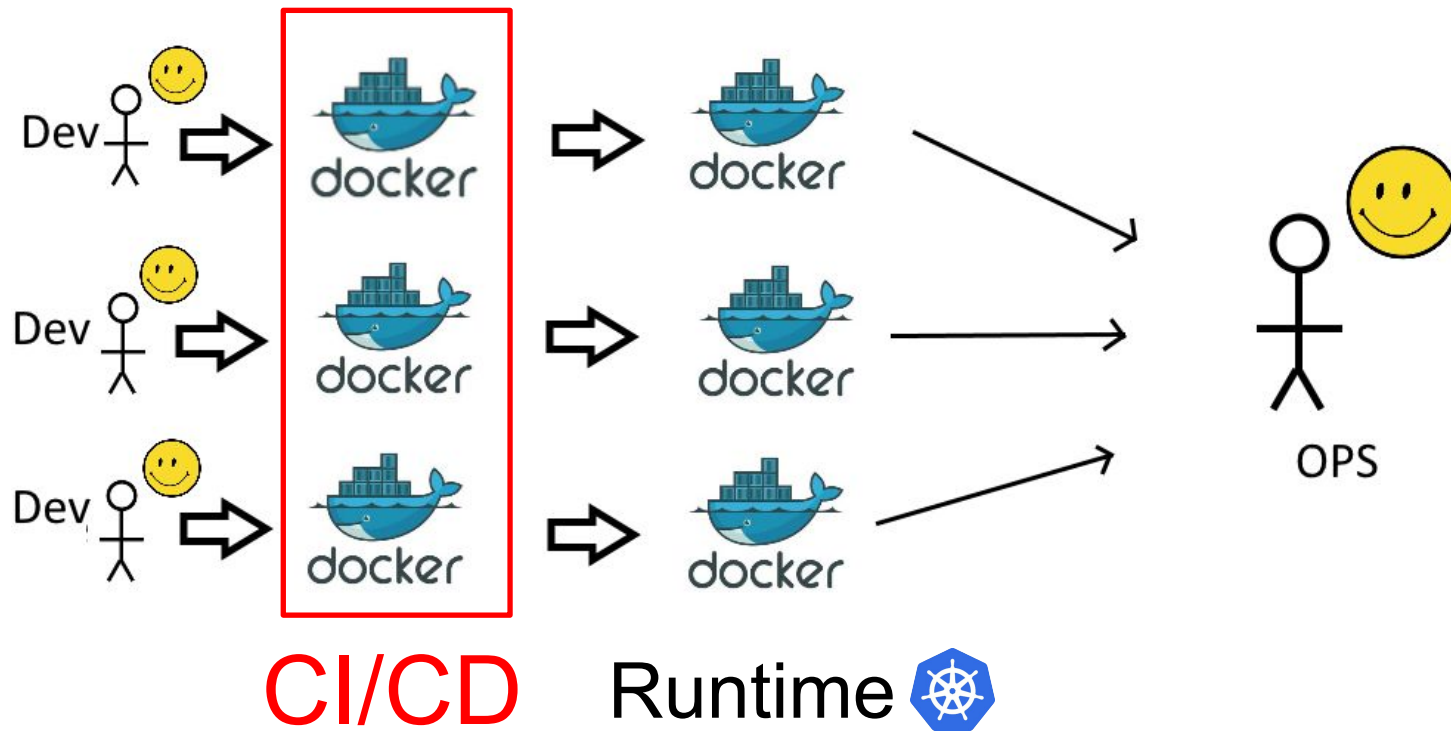


# Adding Docker-based Build Pipelines





# Today's Talk



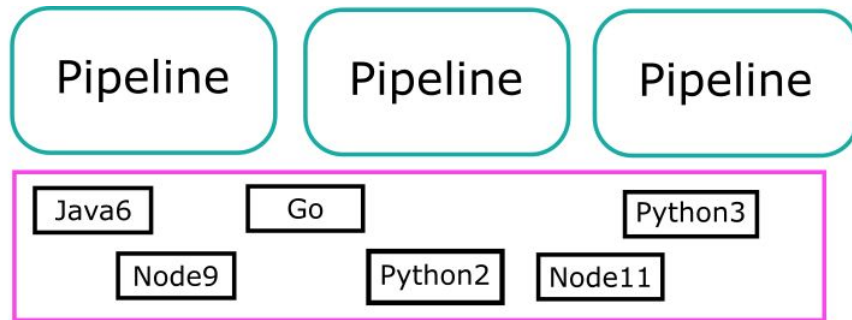
# VM versus Docker

Before:

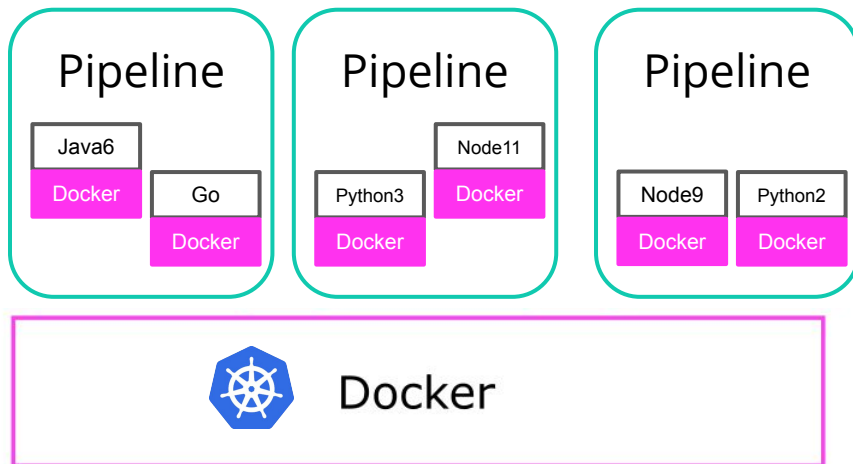
- Tools are static
- Used for all pipelines

After:

- Tools are dynamic
- Isolated to each pipeline



VM



# Resources for Docker as Deployment artifact



Webinars | July 13, 2018

**Skip Staging! Test Docker, Helm, and Kubernetes Apps like a Pro**



Taryn Jones

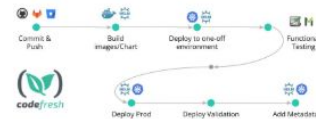


Webinars | June 27, 2018

**Canary Deployment with Helm, Istio, and Codefresh**



Dan Garfield

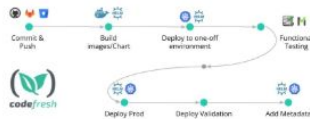


Webinars | May 27, 2018

**Selenium Testing your Kubernetes apps with Machine Learning and Testim**



Dan Garfield



Webinars | May 24, 2018

**Selenium Testing your Kubernetes apps with Machine Learning and Testim**



Taryn Jones

Webinar  
CD for  
Kubernetes  
Apps with  
Helm and  
ChartMuseum



Webinars | May 10, 2018

**Continuous Delivery for Kubernetes Apps with Helm & ChartMuseum**



Taryn Jones



Webinars | May 2, 2018

**New in Codefresh: Single-Sign-On, Custom Triggers, Cron Jobs, and More!**

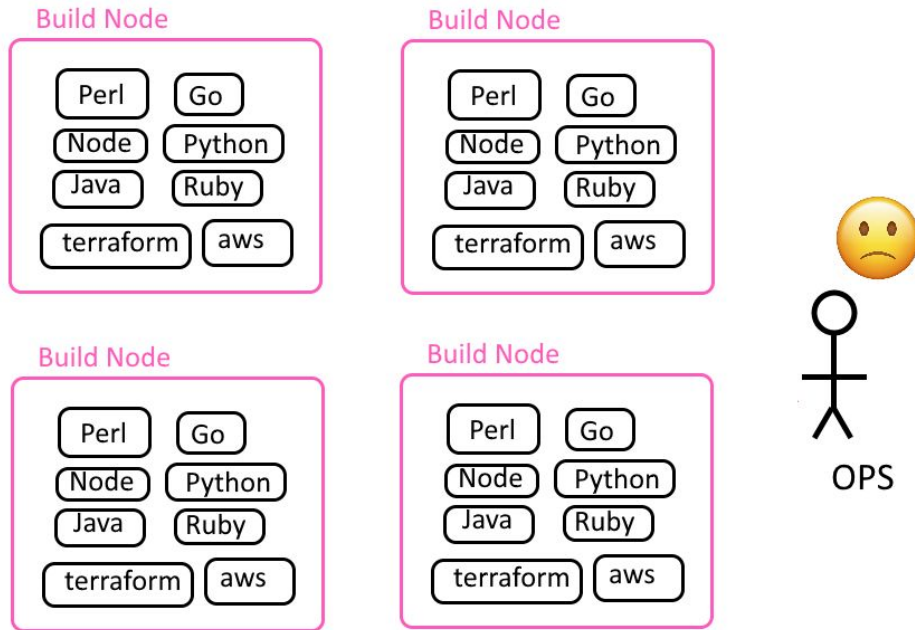


Taryn Jones

# Docker CI/CD - Benefits

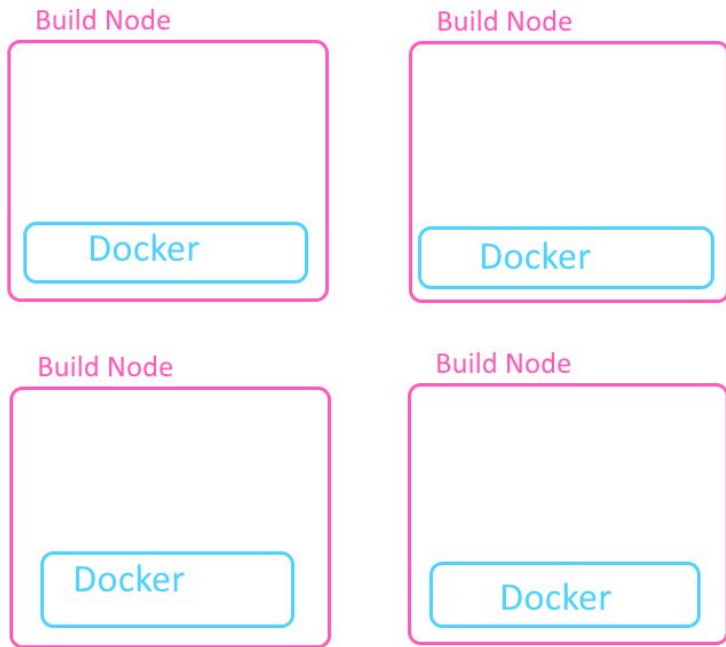
# VM-Based Pipelines

- Multiple tools on each node
- Very hard to manage
- Often nodes had different versions of the same tool
- Developers had to choose the correct machine for their build




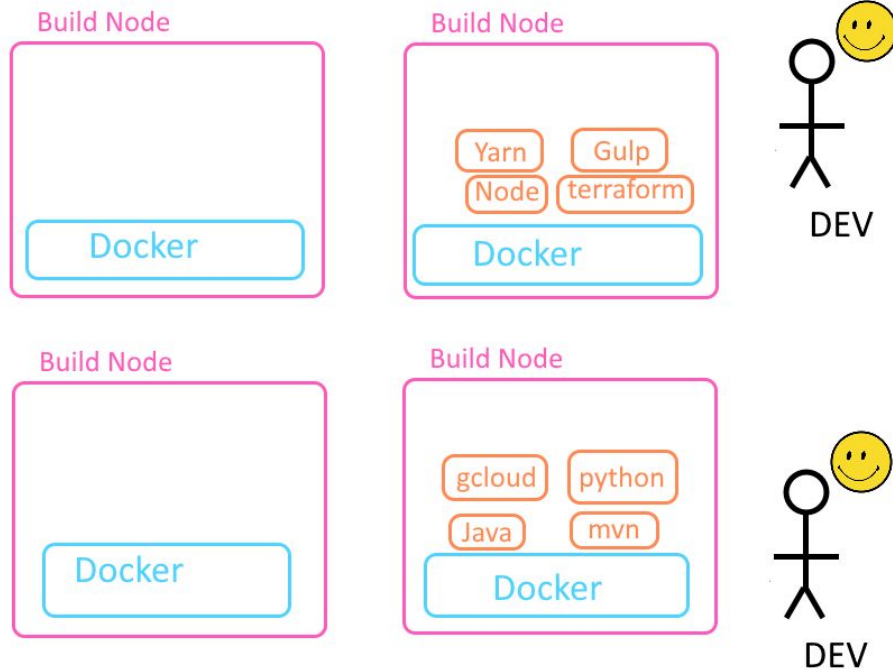
# Build farm after Docker

- Only Docker is installed
- Very easy to manage
- All nodes are exactly the same
- These nodes are often a Kubernetes cluster



# Using Docker in Continuous Integration

- EVERY build tool is placed in a Docker container
- The build node has only Docker installed and nothing else
- A pipeline is a series of commands that run inside a Docker context
- After each build the node reverts back to its original state
- Developers don't care about nodes 



# Docker images are everywhere

- They are reusable and shareable
- No need to reinvent the wheel (e.g. Terraform in Docker)
- Private docker images can be created with your team in mind



# Container per build step

- Codefresh requires ALL tools to be dockerized
- You can use any public or private Docker image as tooling
- Each build step has a Docker image as context
- Pipelines are described in declarative YAML

## Docker Build Context

node:9-alpine

aws-cli:latest

cf-helm:2.9

## Codefresh Pipeline

Step  
npm install  
npm test



Step  
aws s3 sync . s3://mybucket



Step  
push chart

# About Codefresh

- Docker based CI/CD solution
- Each build step is a Docker image
- Native support for Docker, Helm, Kubernetes deployments
- Includes built-in Docker registry and Helm repository
- 30,000+ users

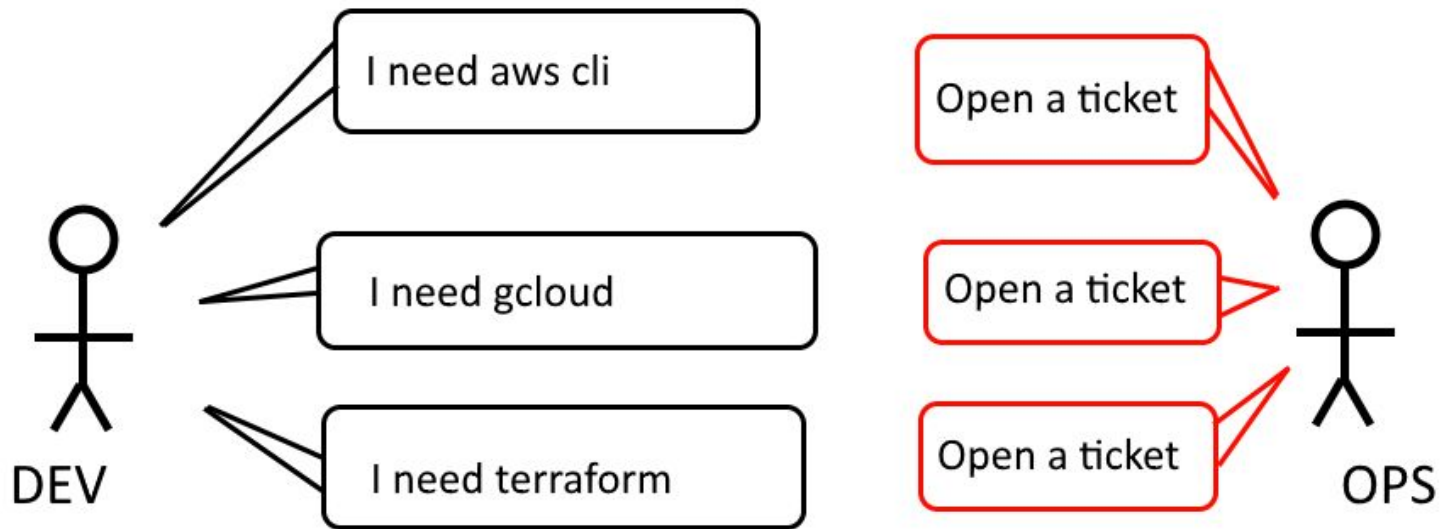
The screenshot displays the Codefresh web interface for a pipeline named "Pipeline Name". The interface includes a sidebar with navigation options: BUILD (PIPELINES, Builds), MONITORING (Kubernetes, Helm Releases, Docker Swarm), RESOURCES (Images, Repositories, Helm Charts), and CONFIGURATION (Account Settings, User Settings). The main area shows the pipeline execution details, including a "COMPLETED" status, 12 steps, a start time of 1/8/2018 22:22, a duration of 10m, and a trigger of "COMMIT on" by "Man's Gitlab - codefresh-io/secrets" by "Man Arbel". The pipeline is visualized as a sequence of steps grouped into three phases: "Initialization", "BUILD", and "UNIT". Each phase contains a list of steps, all of which are "GET CLONE Cloning main repository" and have a duration of 2.43s. A "Restart from Step" button is visible over the first step of the BUILD phase. The interface also includes links for "Documentation", "Support", and "TRIGGER PIPELINE", and a "Download Log" button.

# Demo 1:

# Python/Node application

[https://github.com/containers101/docker-based-pipelines-webinar/tree/master/01\\_simple\\_pipeline](https://github.com/containers101/docker-based-pipelines-webinar/tree/master/01_simple_pipeline)

# Traditional VM based problems



# VM based Platform Questions:

- Do you support my favorite version of Node/Java/Go/Ruby/Python?
- Do you support maven, yarn, gulp, sbt, gradle, rake?
- Can I run Ansible? Terraform? GCloud? AWS CLI?
- Can I run Kubectl? Helm? Draft?



# VM based CI/CD Platforms

## Use PHP with updated curl version #9924



Nyholm opened this issue on Jul 29 · 1 comment

## Add Python 3.7 option #9815



Harmon758 opened this issue on Jun 28 · 73 comments

## scala / sbt 1.x support #9816



aryairani opened this issue on Jun 28 · 9 comments

## Upgrades

Ansible gets an update with version [2.6.1](#).

ChromeDriver is now update to version [2.40](#).

Docker Compose has been updated to version [1.22.0](#).

Elixir gets a version update with [1.6.6](#).

Gecko dirver is now on version [0.21.0](#).

Google Chrome is updated to version [67.0.3396.99](#).

Go receives two updates with [1.9.7](#) and [1.10.3](#).

Git has been updated to version [2.18.0](#).

Java gets three updates with [7u181](#), [8u181](#) and [10.0.2](#).

Maven gets an update with version [3.5.4](#).

MongoDB has been updated to version [3.11](#).

NodeJS receives an update with version [8.11.3](#).

PHP gets two updates with [7.1.9](#), [7.2.5](#).



# Demo 2:

# Adding Go and AWS CLI

[https://github.com/containers101/docker-based-pipelines-webinar/tree/master/02\\_aws\\_cli](https://github.com/containers101/docker-based-pipelines-webinar/tree/master/02_aws_cli)



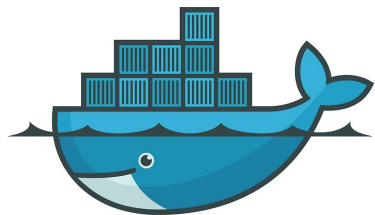
## Does Codefresh Support...

- Node 10?
- Perl 6?
- Python2?
- Gradle?
- Vault?
- AWS cli?
- Sonar?
- Findbugs?
- Selenium?
- Snyk?
- Clair?



# YES!

Because there is a Docker  
image for it



docker

## Does Codefresh Support...

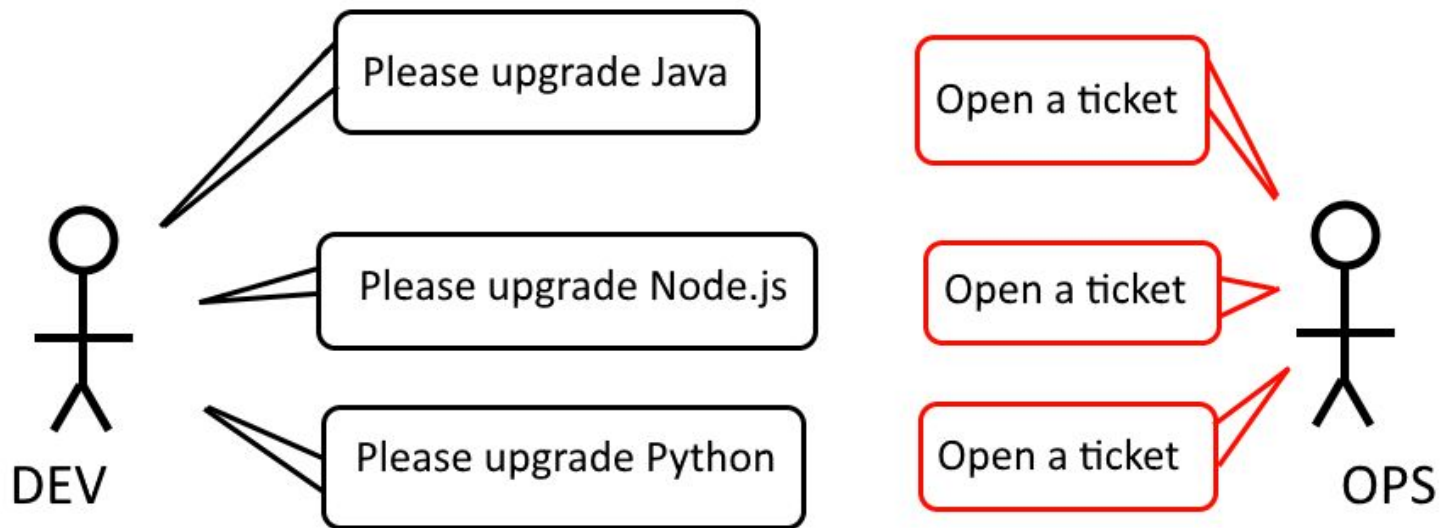
- Node 10?
- Perl 6?
- Python2?
- Gradle?
- Vault?
- AWS cli?
- Sonar?
- Findbugs?
- Selenium?
- Snyk?
- Clair?

# Codefresh Pipelines are Future Proof

- You can use ANY existing Docker image from Dockerhub or any other Registry
- Every time a new tool comes out, it can be used right away if packaged in a Docker image

# Tool Upgrades and Version Clashes

# Updating a Tool in a VM based Pipeline



# VM based CI Solutions

Please add PHP 7.3 images #9717

 Open Majkl578 opened this issue on Jun 8 · 47 comments

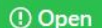
Upgrade to Maven 3.5.3 #9366

 Open vincent-zurczak opened this issue on Mar 19 · 7 comments

C++14, Qt5.7 #6503

 Open mrdeveloperdude opened this issue on Aug 19, 2016 · 12 comments

Support for pypy/pypy3 v6.0+ python #9542

 Open webknjaz opened this issue on Apr 26 · 4 comments

older versions of R no longer available? #9751

 Open achubaty opened this issue on Jun 15 · 4 comments

Update Git #6328

 Open joepvd opened this issue on Jul 18, 2016 · 31 comments

How can I upgrade Python to the latest 2 version? (2.7.15) #10273

 Open lipis opened this issue 21 days ago · 3 comments

**OBSOLETE**

# Demo: Updating Python to 3.7

[https://github.com/containers101/docker-based-pipelines-webinar/tree/master/02\\_aws\\_cli](https://github.com/containers101/docker-based-pipelines-webinar/tree/master/02_aws_cli)

# Using Tools from Different Versions

- Version clashes are a huge pain for both developers and operators
- Legacy projects need to still use old version
- Using different versions in the same pipeline is almost impossible
- Developers want to use latest version of tool, traditional CI/CD platforms may not be able to keep up

# Wasting Effort on “Version Managers”

## Ruby Version Manager (RVM)

RVM is a command-line tool which allows you to easily install, manage, and work with multiple ruby environments from interpreters to sets of gems.



## Node Version Manager

build passing

version v0.33.11

cii best practices passing

### Table of Contents

- Installation
  - Install script
  - Verify installation
  - Important Notes
  - Git install
  - Manual Install
  - Manual upgrade

## Simple Python Version Management: pyenv

gitter join chat

build passing

pyenv lets you easily switch between multiple versions of Python. It's simple, unobtrusive, and follows the UNIX tradition of single-purpose tools that do one thing well.

This project was forked from [rbenv](#) and [ruby-build](#), and modified for Python.

**OBSOLETE**



# Wasting Effort on “Version Managers”

- They allow developers to switch between different versions
- Tied to a specific technology/programming language
- Require their own installation/ maintenance
- Must be upgraded for new versions

Node Version Manager build passing version v0.33.11 ci test passing

## Table of Contents

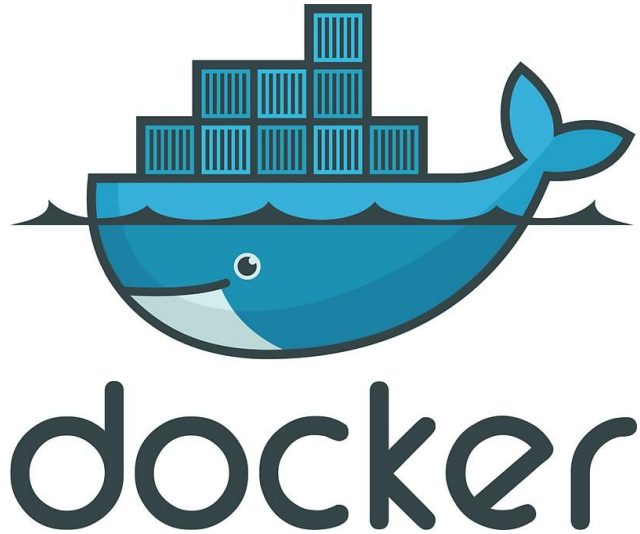
- Installation
  - Install script
  - Verify installation
  - Important Notes
  - Git install
  - Manual Install
  - Manual upgrade

**OBSOLETE**

```
language: python
python:
  - "2.6"
  - "2.7"
  - "3.3"
  - "3.4"
  - "3.5"
  - "3.5-dev" # 3.5 development branch
  - "3.6"
  - "3.6-dev" # 3.6 development branch
  - "3.7-dev" # 3.7 development branch
# command to install dependencies
install:
  - pip install -r requirements.txt
# command to run tests
script:
  - pytest
```

# The Problem with Python

- Different python versions are a notorious problem
- Until recently you needed dedicated support from your CI platform
- What happens if I want to test Python 2.5?



## Replacing “version managers” with Docker

- Works for any language/framework
- Already installed on the build node
- Its own version is independent from the tools
- Can use any public and private image

# Codefresh “Python Support”

- We support EVERY container ever made
- We support EVERY container that you can make in the future

- 3.5.6-alpine3.8 , 3.5-alpine3.8 , 3.5.6-alpine , 3.5-alpine ([3.5/alpine3.8/Dockerfile](#))
- 3.5.6-alpine3.7 , 3.5-alpine3.7 ([3.5/alpine3.7/Dockerfile](#))
- 3.4.9-stretch , 3.4-stretch ([3.4/stretch/Dockerfile](#))
- 3.4.9-slim-stretch , 3.4-slim-stretch , 3.4.9-slim , 3.4-slim ([3.4/stretch/slim/Dockerfile](#))
- 3.4.9-jessie , 3.4-jessie ([3.4/jessie/Dockerfile](#))
- 3.4.9-slim-jessie , 3.4-slim-jessie ([3.4/jessie/slim/Dockerfile](#))
- 3.4.9-wheezy , 3.4-wheezy ([3.4/wheezy/Dockerfile](#))
- 3.4.9-alpine3.8 , 3.4-alpine3.8 , 3.4.9-alpine , 3.4-alpine ([3.4/alpine3.8/Dockerfile](#))
- 3.4.9-alpine3.7 , 3.4-alpine3.7 ([3.4/alpine3.7/Dockerfile](#))
- 2.7.15-stretch , 2.7-stretch , 2-stretch ([2.7/stretch/Dockerfile](#))
- 2.7.15-slim-stretch , 2.7-slim-stretch , 2-slim-stretch , 2.7.15-slim , 2.7-slim , 2-slim ([2.7/stretch/slim/Dockerfile](#))
- 2.7.15-jessie , 2.7-jessie , 2-jessie ([2.7/jessie/Dockerfile](#))
- 2.7.15-slim-jessie , 2.7-slim-jessie , 2-slim-jessie ([2.7/jessie/slim/Dockerfile](#))
- 2.7.15-wheezy , 2.7-wheezy , 2-wheezy ([2.7/wheezy/Dockerfile](#))
- 2.7.15-alpine3.8 , 2.7-alpine3.8 , 2-alpine3.8 , 2.7.15-alpine , 2.7-alpine , 2-alpine ([2.7/alpine3.8/Dockerfile](#))
- 2.7.15-alpine3.7 , 2.7-alpine3.7 , 2-alpine3.7 ([2.7/alpine3.7/Dockerfile](#))
- 2.7.15-alpine3.6 , 2.7-alpine3.6 , 2-alpine3.6 ([2.7/alpine3.6/Dockerfile](#))
- 2.7.15-windowsservercore-ltsc2016 , 2.7-windowsservercore-ltsc2016 , 2-windowsservercore-ltsc2016 ([2.7/windows/windowsservercore-ltsc2016/Dockerfile](#))
- 2.7.15-windowsservercore-1709 , 2.7-windowsservercore-1709 , 2-windowsservercore-1709 ([2.7/windows/windowsservercore-1709/Dockerfile](#))

# Demo 3:

# Multiple Python/Node versions

[https://github.com/containers101/docker-based-pipelines-webinar/tree/master/03\\_multiple\\_versions](https://github.com/containers101/docker-based-pipelines-webinar/tree/master/03_multiple_versions)

# Data Sharing Between Pipeline Steps

# Data Sharing

- Steps need to communicate
- Output of one step is input for the next
- Artifacts (node modules, ruby gems, maven caches) need to persist
- Test reports/Coverage statistics

## Docker Build Context

maven:3.5.2

terraform:light

cf-deploy-ecs:1.0

docker-lftp:1.0

## Codefresh Pipeline

mvn package  
Step

tf apply ...  
Step

cfecs-update ...  
Step

lftp ..  
Step

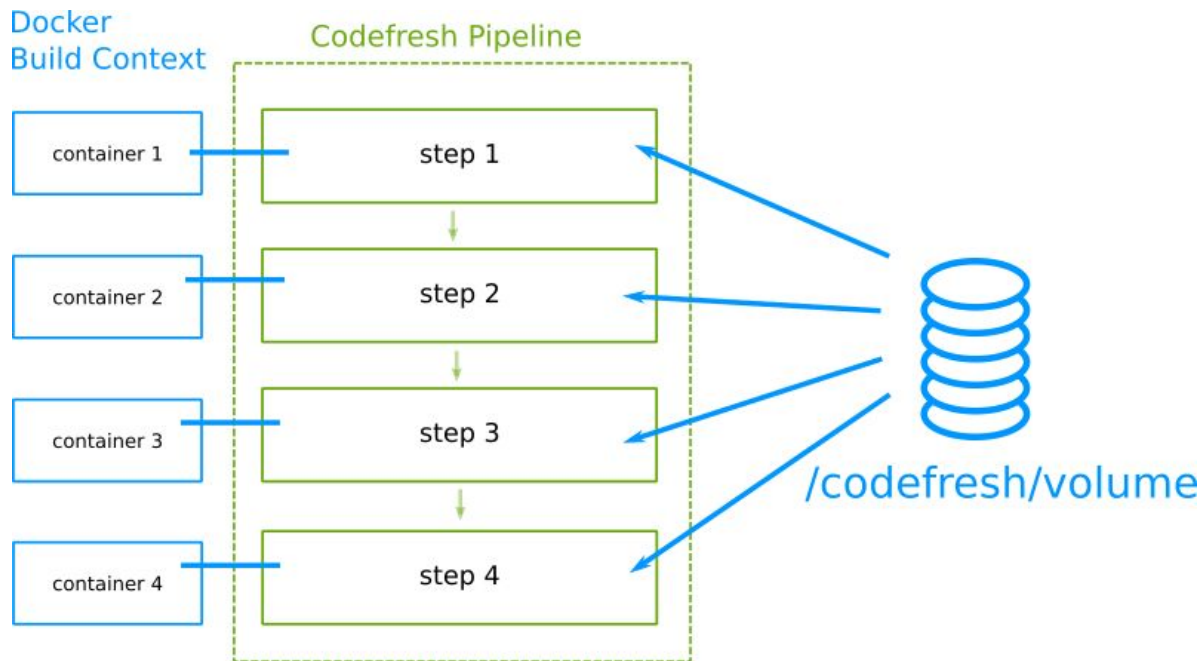
# Caches and Artifacts (VM based solutions)

- “Cache” directive
  - Need to be setup explicitly
  - Different for each build tool
- “Artifact” directive
  - Developers defines exact path of what needs to be archived
  - Used for the result of the whole build or as shared data between steps





# All Steps Share a Volume in Codefresh



# Project is on the Volume

- Project is checked out in the volume
- Volume is also persisted between builds
- Any build tools that use the project folder for artifacts will gain automatic caching
- For other tools you just need to point their cache to /codefresh/volume
- There is no need for special “artifact settings”. Just place files in /codefresh/volume

/codefresh/volume



my-git-project



file 1



file 2



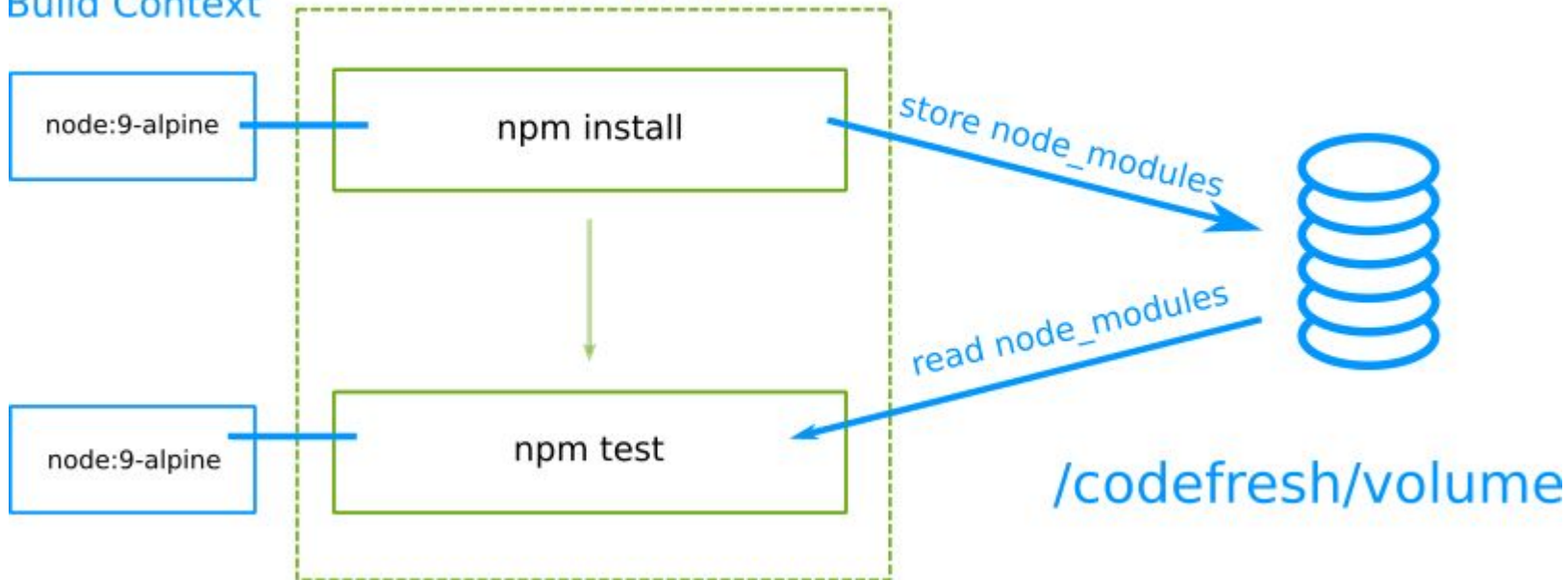
file 2

Default  
Working directory



# Demo 4 – Node Modules

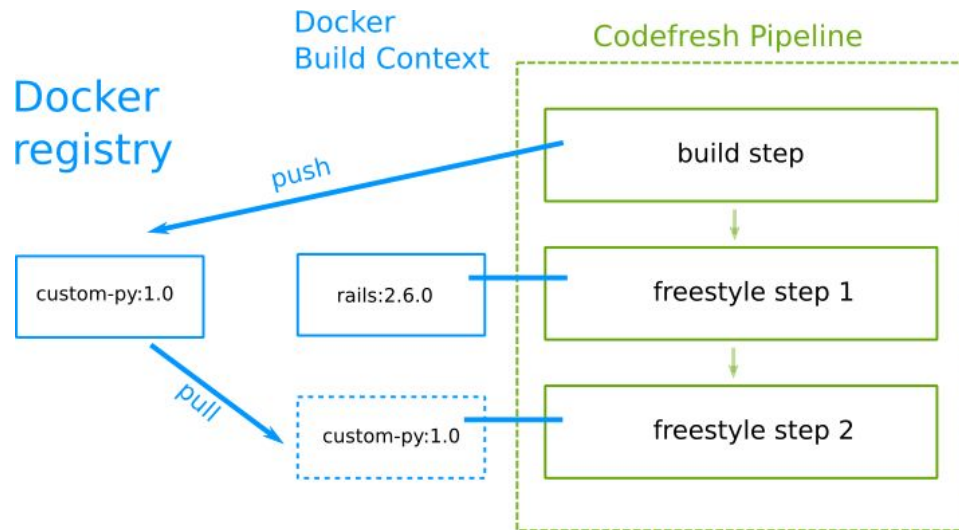
Docker  
Build Context



[https://github.com/containers101/docker-based-pipelines-webinar/tree/master/04\\_volume](https://github.com/containers101/docker-based-pipelines-webinar/tree/master/04_volume)

# Creating Docker Images On-demand

- Create a Docker image as a step
- Use image in a later step
- Maximum flexibility for build context
- Image contents are not known in advance
- Codefresh is the only platform at the moment that offers this capability



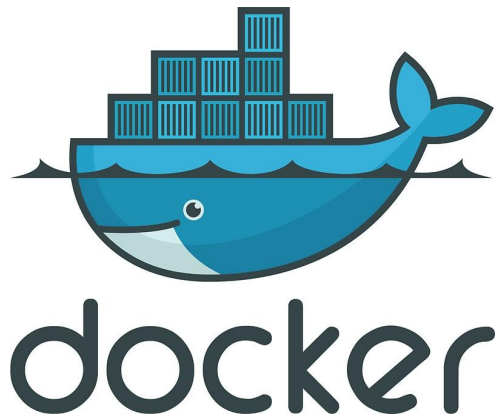
# Codefresh Plugins

# Plugins in Traditional CI/CD Platforms

- Specific to the platform (vendor lock-in)
- Tied to a specific language (e.g. Groovy)
- Developer needs to learn proprietary API
- Testing and installing them is difficult

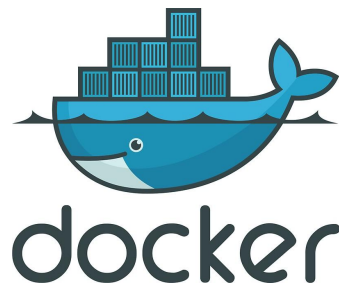


# Codefresh Plugins = Docker Images



# Codefresh Plugins

- Not tied to any programming language
- Require only Docker knowledge
- Easy to test, easy to search, easy to store
- Several plugins for Codefresh already available





# Plugin Directory

<http://steps.codefresh.io/>

The screenshot shows the Codefresh Plugin Directory. On the left, there's a sidebar with 'CATEGORIES' including Code Review, Command Line, Deployment, Kubernetes, Messaging, Publishing, Security, and Utility. Below this is a 'Submit a Step' button. The main area is titled 'FEATURED' and displays a grid of 15 plugin cards. Each card has an icon, a title, a brief description, and a 'VIEW' link. The plugins include: Annotate GitLab Merge Requests, Blue/Green Deploy, Build using Azure ACR, Canary Deployment, Codefresh Kompose Plugin, Copy to S3, Create GKE Cluster, Create Helm Chart, Create an Anchore Release, Deploy Serverless Service, Deploy to Amazon ECS, Deploy to DC/OS clusters, Import Docker Images, Interact with Jira, and Manage Bintray Releases.



## Blue/Green Deploy

Perform blue/green deployments on a Kubernetes cluster

[github.com](#)

Kostis Kape...

12/12/2018

DEPLOYMENT

KUBERNETES

Copy to Clipboard

```
type: freestyle
image: codefresh/sample

blueGreenDeploy:
  title: "Deploying new version ${CF_SHORT_REVISION}"
  image: "codefresh/k8s-blue-green:master"
  environment:
    - "SERVICE_NAME=my-demo-app"
    - "DEPLOYMENT_NAME=my-demo-app"
    - "NEW_VERSION=${CF_SHORT_REVISION}"
    - "HEALTH_SECONDS=60"
    - "NAMESPACE=colors"
    - "KUBE_CONTEXT=myDemoAKSCluster"

  next_step:
    type: build
```

VARIABLES

Copy to Clipboard

➔ **KUBE\_CONTEXT** - The name of your cluster (as found in the Codefresh Kubernetes dashboard).

# Learn how to build your own!

Workshop

[github.com/  
todaywasawesome/  
containers-as-steps](https://github.com/todaywasawesome/containers-as-steps)

# Summary

- Docker-based pipelines use Docker images as build steps
- Upgrading tools is easy
- Using multiple versions of the same tool is trivial
- Can dynamically create build steps
- Codefresh plugins are Docker images



# Thank You!

**Build Fast,  
Deploy Faster**

Signup for a FREE account with  
UNLIMITED builds

& Schedule a 1:1 with  
our experts at  
[codefresh.io](https://codefresh.io)

[guy.salton@codefresh.io](mailto:guy.salton@codefresh.io)

