DWARVES FOUNDATION

Elixir Deployment

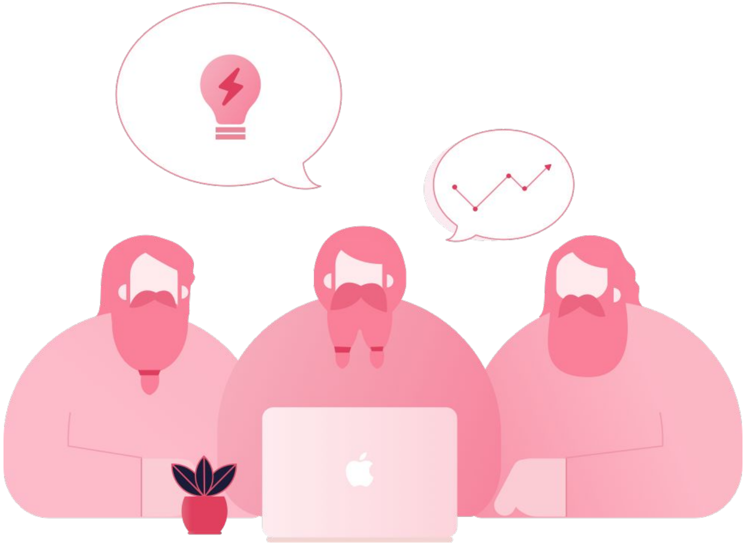
Efficient Strategies for Deploying Elixir Applications

Hieu Phan

Dwarves Foundation

Backend Lead

andy@dwarvesv.com



Agenda

Explore efficient Elixir deployment strategies: from release management to CI/CD integration.

1. Why?
2. Deployment in Elixir
3. Demo

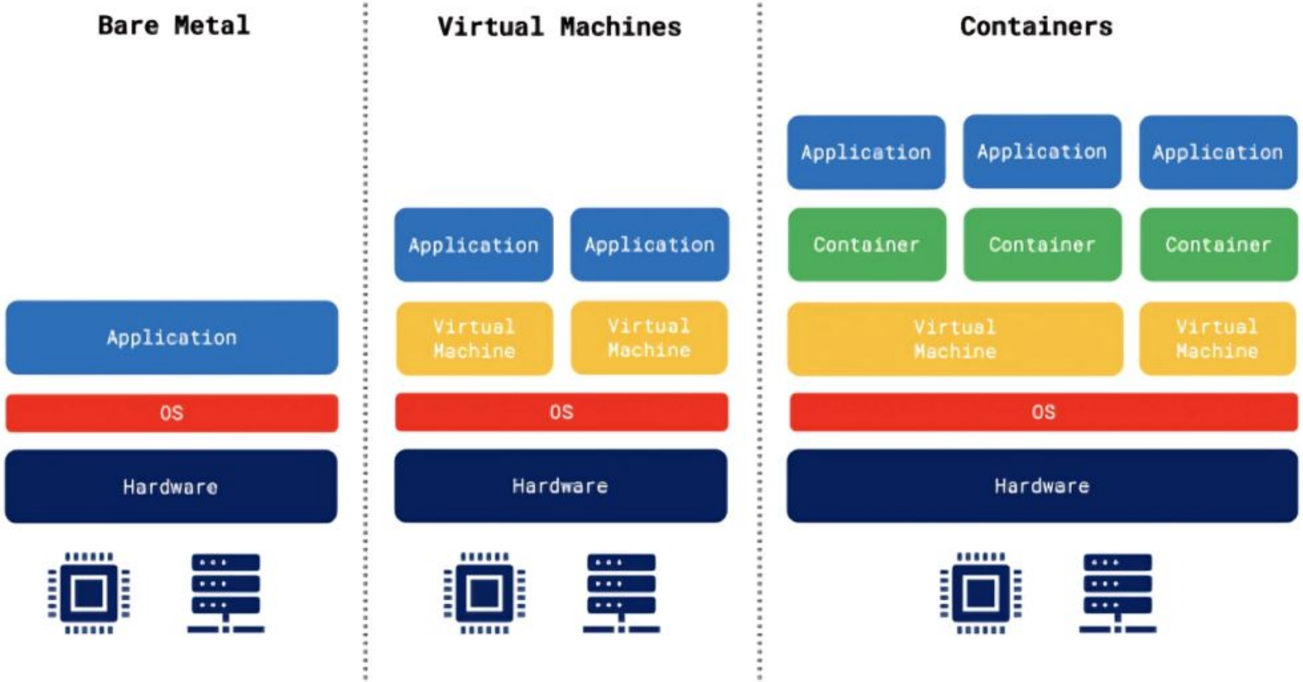
Why?

Deployment in Elixir is essential for several reasons

Why?

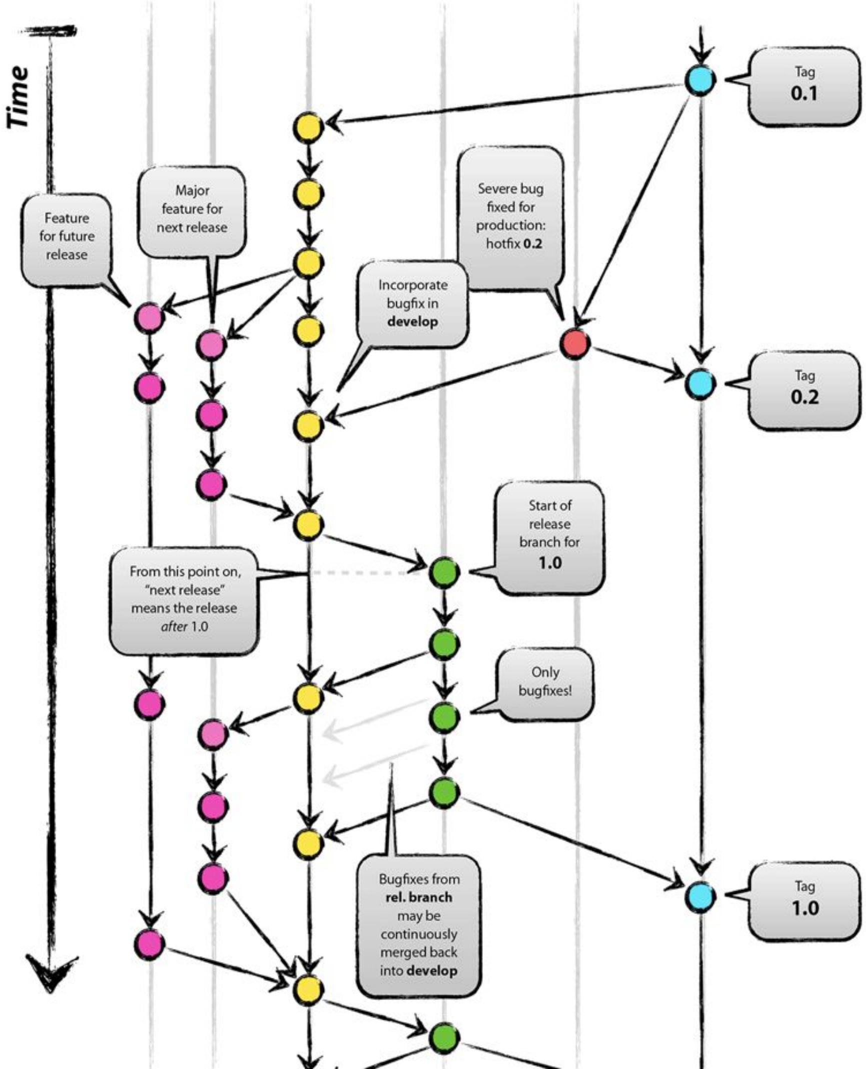
- Product Delivery: CI/CD
- Developer Experience
- Scalability, Reliability and Stability
- …

History


Bare Metal
Virtual Machines
Containers
Application
OS
Hardware
Application
Application
Virtual Machine
Virtual Machine
OS
Hardware
Application
Application
Application
Container
Container
Container
Virtual Machine
Virtual Machine
OS
Hardware

CI/CD

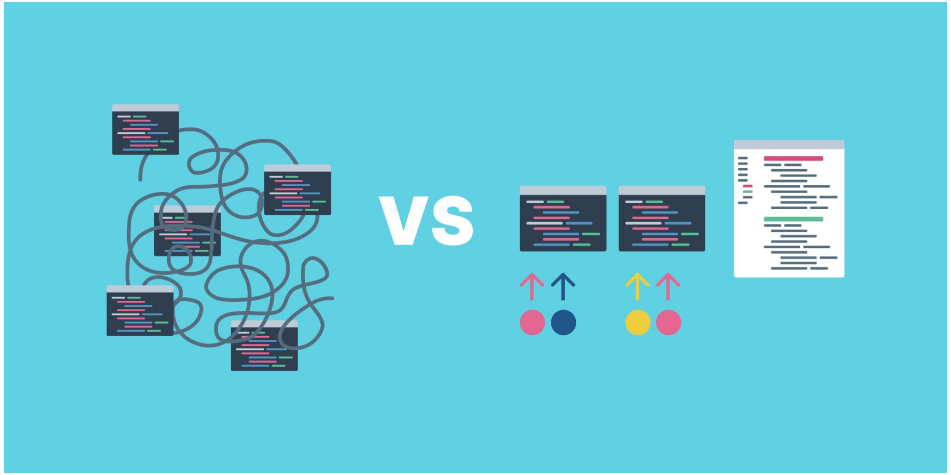
- Development
- Testing
- CI/CD
- Deployment


Time
Tag 0.1
Major feature for next release
Feature for future release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

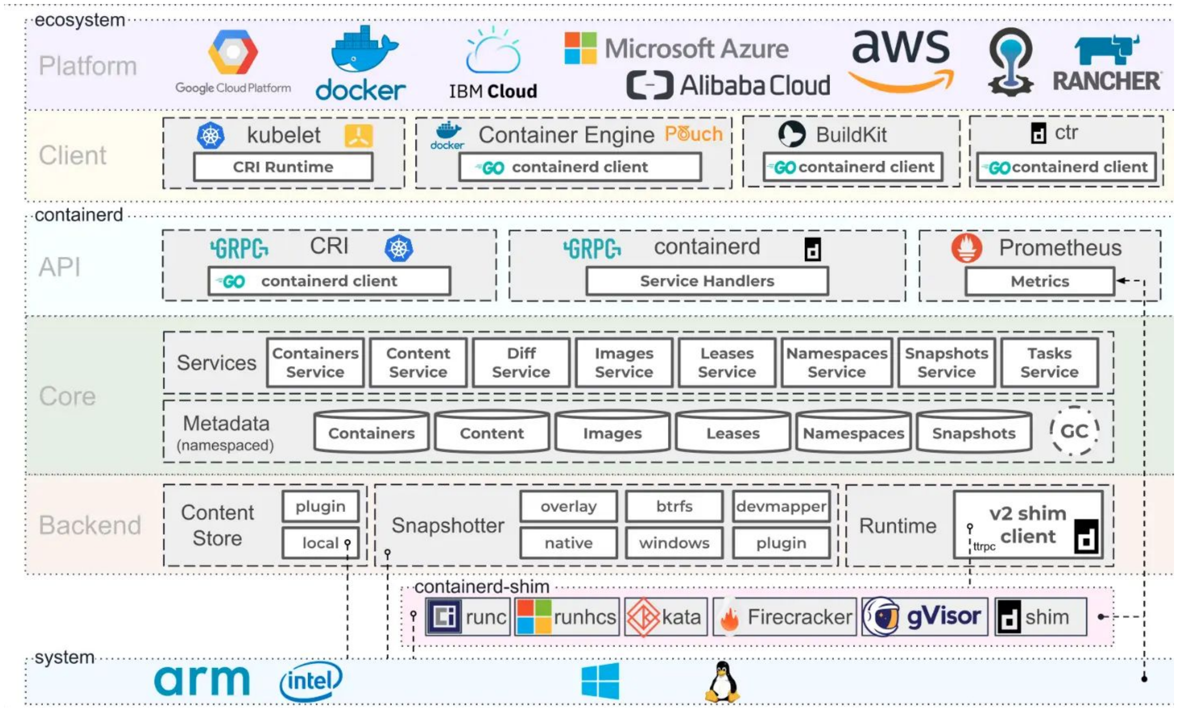
DWARVES FOUNDATION

Developer Experience

- Isolation and Consistency
- Automation
- Onboarding Member Cost
- Handover



Real life


ecosystem
Platform
Google Cloud Platform
docker
IBM Cloud
Microsoft Azure
Alibaba Cloud
aws
RANCHER
Client
kubelet
CRI Runtime
docker
Container Engine
Pouch
containerd client
BuildKit
containerd client
ctr
containerd client
containerd
API
GRPC
CRI
containerd client
GRPC
containerd
Service Handlers
Prometheus
Metrics
Core
Services
Containers Service
Content Service
Diff Service
Images Service
Leases Service
Namespaces Service
Snapshots Service
Tasks Service
Metadata (namespaced)
Containers
Content
Images
Leases
Namespaces
Snapshots
GC
Backend
Content Store
plugin
local
Snapshotter
overlay
btrfs
devmapper
native
windows
plugin
Runtime
v2 shim client
ttrpc
containerd-shim
runc
runhcs
kata
Firecracker
gVisor
shim
system
arm
intel

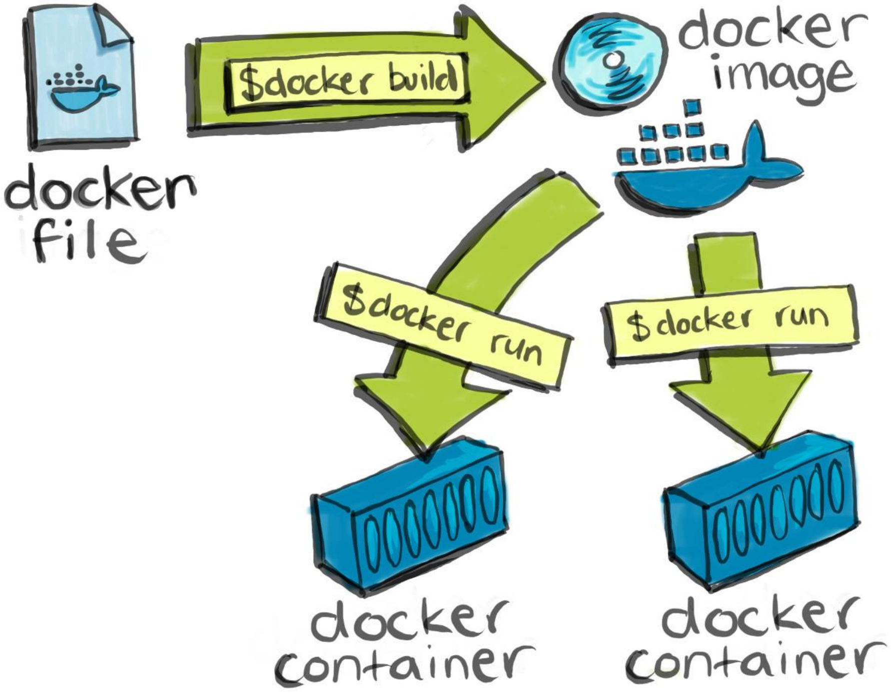
Deployment in Elixir

Walk through from start to finish the deployment in

Elixir

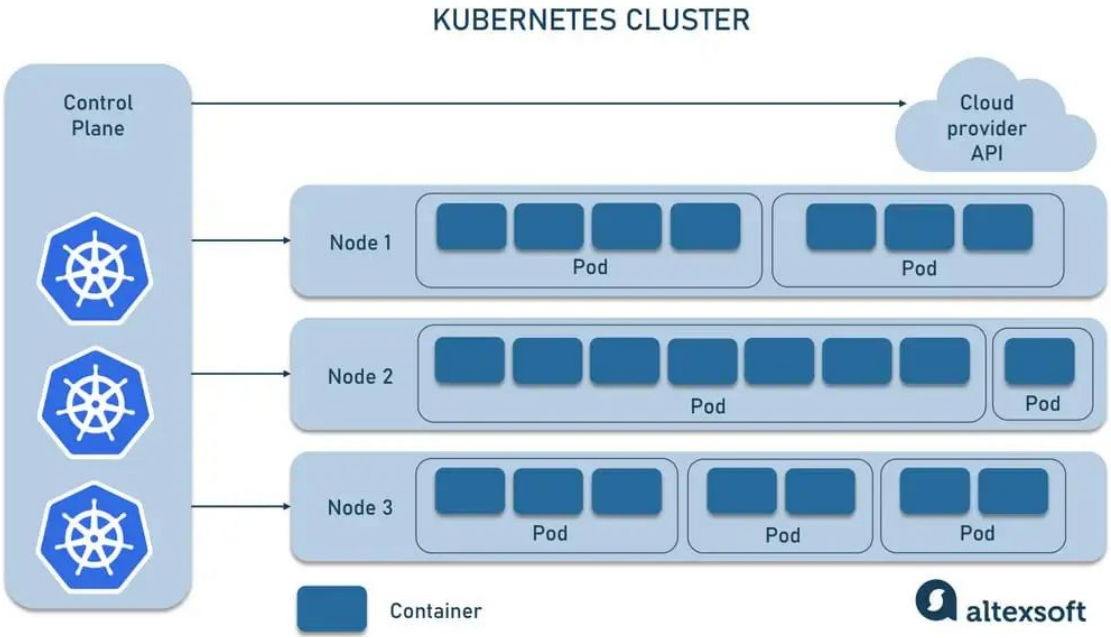
Docker working flow

- Dockerfile
- Docker image
- Docker container
- Variable environment
- Network
- Storage



K8s

- Master
- Node
- Cluster

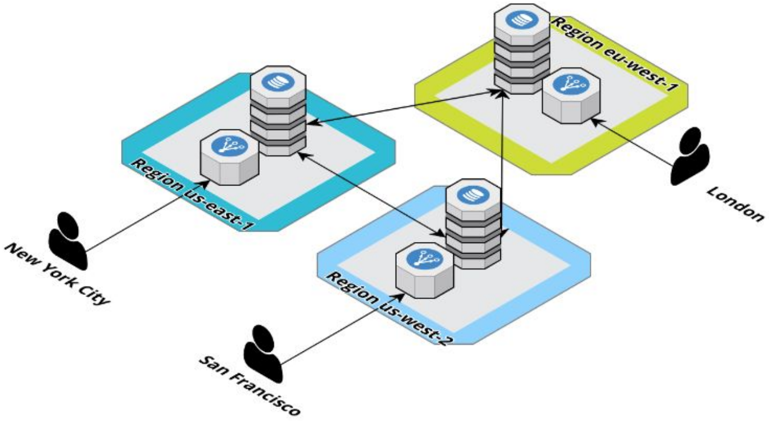


Basic

● Project type: umbrella? single app	● mix new <app> --sup --umbrella
● Dockerfile	● touch Dockerfile
● Build Image	● docker build -t <image_name> .
● Config & Environment	●
● Deploy to docker	● docker run -d --name container-name -p 4000:4000 <image_name>

NOT Basic

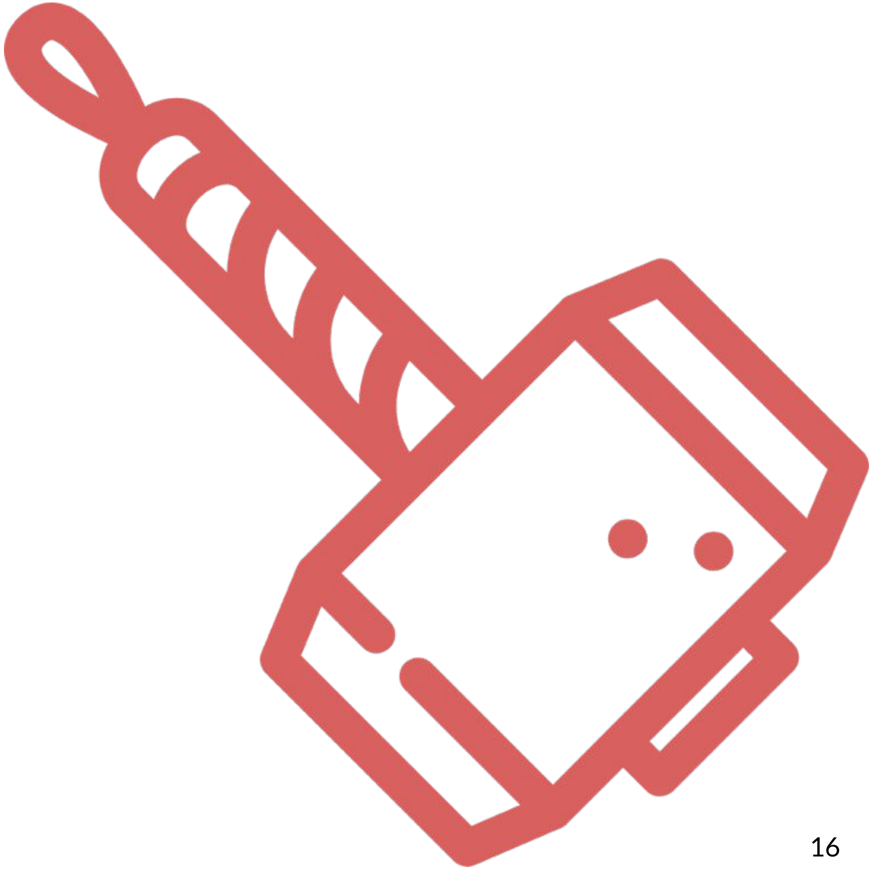
- Cluster contains many containers
- Containers collaboration
- Multi-Region Infrastructure Deployment



Demo

Demo - Zer0 to Hero

- New a project
- Prepare Dockerfile
- Build a image
- Push the image to a register
- Deploy the image locally



Reference

Resources & Reference links

- https://www.docker.com/
- https://kubernetes.io/
- https://github.com/bitwalker/distillery
- https://hexdocs.pm/mix/Mix.Tasks.Release.html

Thank You

DWARVES
FOUNDATION

Q&A