With WasmEdge to New Shores

Max Körbächer | Co-Founder & Cloud Native Advocate @ Liquid Reply

REPLY
LIQUID

Say hi!

Max Körbächer - Co-Founder of Liquid Reply

My work is all about
Kubernetes Consultancy & Cloud Native Advisory

- Former Enterprise Architect, yet design and build hyper converged infrastructures and cloud agnostic solutions
- Contributing to the Kubernetes release team, related K8s technologies and Co-Chair of the CNCF Environmental Sustainability Working Group

mkoerbaecher

mkoerbi

Docker has changed the game

Docker/Container changed:

- the way we **design** and **build** applications
- caused a whole **ecosystem** with hundreds of open source systems to appear
- drive adoption from all kind of **cloud provider**
- changed the way we do **automation**
- pushed the development of an **OCI standard**

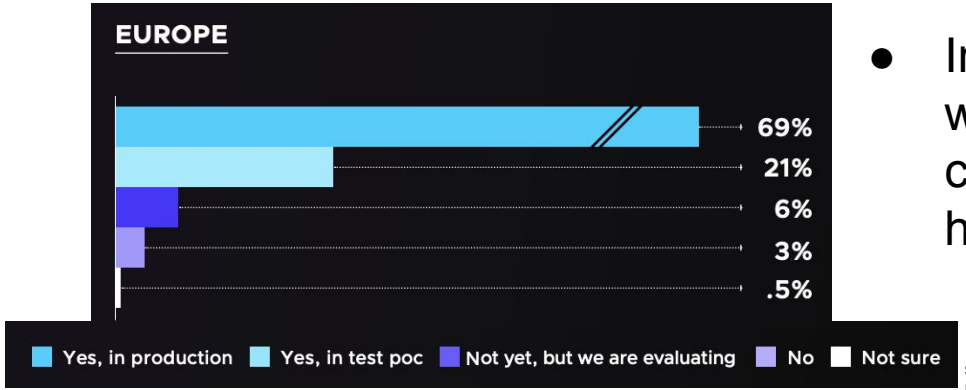
Corporations using or planning to use container

93%

In Containers we trust in Kubernetes we build

- Kubernetes leverages container and got a **defacto standard** for container orchestration (also there are many other nice implementations)

- Kubernetes gets implemented and **used everywhere** (cloud, IaaS, on metal, on egde) - it simplifies a lot, but it also raises the complexity



- In Europe **>= 90%** organizations working with or on K8s - this is comparable to the usage of hypervisor (92%)

src: https://www.cncf.io/wp-content/uploads/2022/02/CNCF-AR_FINAL-edits-15.2.21.pdf
https://www.spiceworks.com/marketing/reports/state-of-virtualization/

Container & Kubernetes

Both together has changed and influenced the ICT world massively

A big bang for a total new market

Boosting open source and a community driven development to new levels

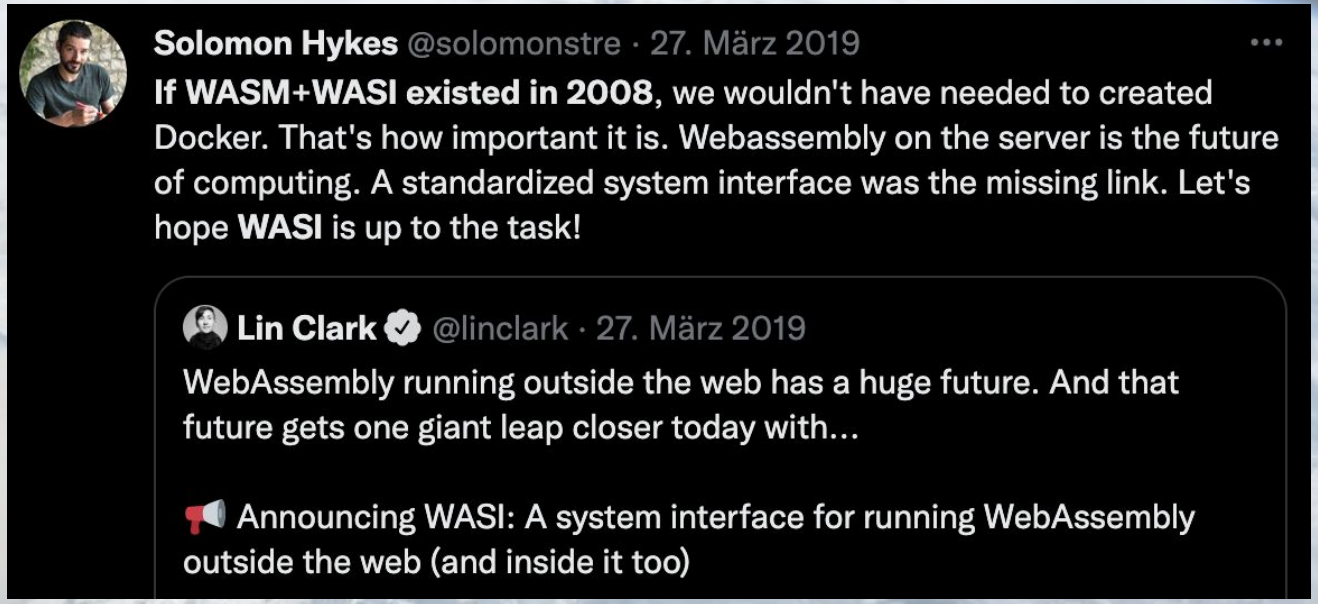
Changed the way we see infrastructure -> Infra as Apps

Security, observability, any kind of extension is seen as a simple plug & play

K8s abstracts away hypervisors, CSP and IaaS

K8s create a knowledge voidness

What next?



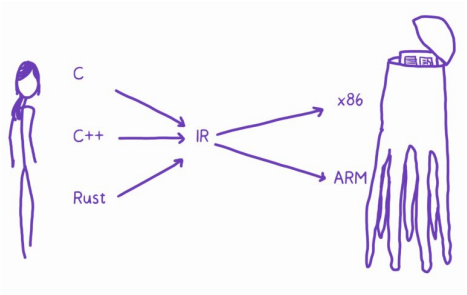
src: https://twitter.com/solomonstre/status/1111004913222324225

What is WebAssembly (WASM)?

Intermediate Layer

Various programming languages and many different execution environments

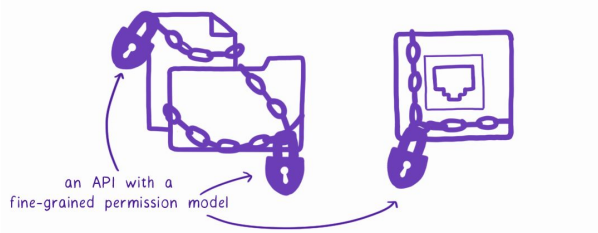
CPU & OS agnostic



Secure

Per default a WASM component is allowed to do nothing

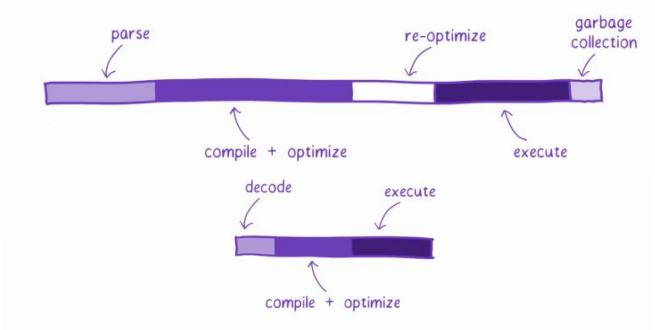
Encapsulated binary, no OS within, nothing to “hack”



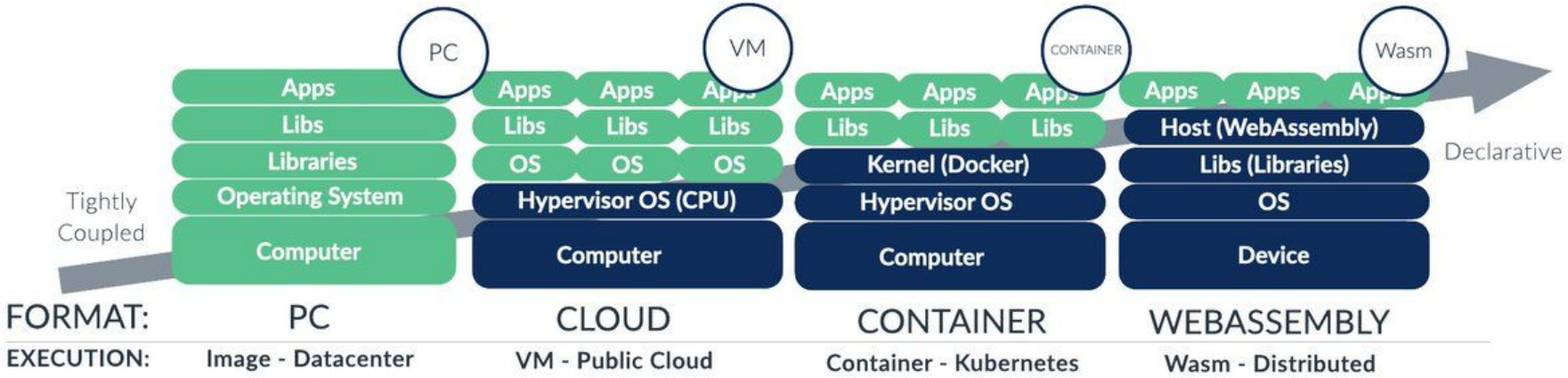
Fast(er)

Drastically short startup time (x100 faster than a container)

Micro footprint, measured in MB not GB



A new paradigm (?)



Where can WebAssembly be applied?

***outside the Browser**

Language Interoperability

Write that library once in a language of your choice; use in any language.

Figma
Lichess.org
Google Earth
Adobe Photoshop

Plugin Systems

Never trust third parties!

Envoy / Istio
Kubewarden
MS Flight Simulator
Minecraft
RedPanda

Embedded Sandboxing

Prevent yourself against bugs of third party libraries.

Firefox
HTTP Servers

Blockchains

Write Smart Contracts in a language of your choice.

CosmWasm
eWASM

Containerisation

Universal Runtime, capability based security model.

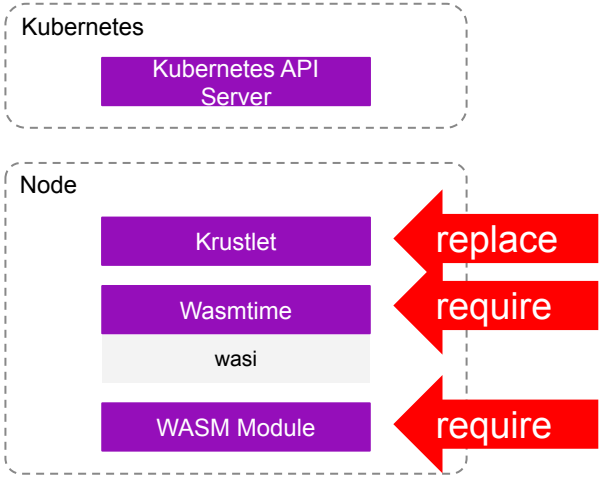
Krustlet
Hippo
wasmCloud
Lunatic
WasmEdge

Serverless Platforms

Minimal Startup time, maximal isolation.

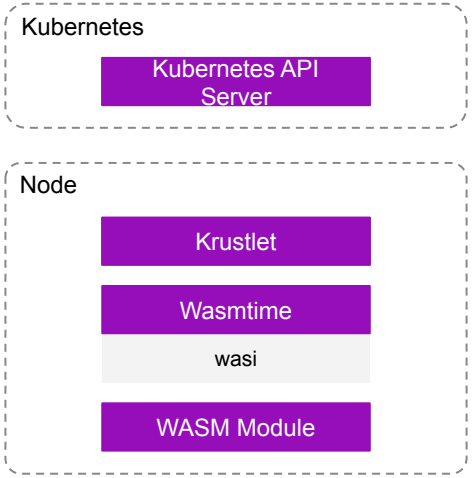
Cloudflare Workers
AWS Lambda
Atmo (Suborbital)
Fastly Compute@Edge

Example implementations

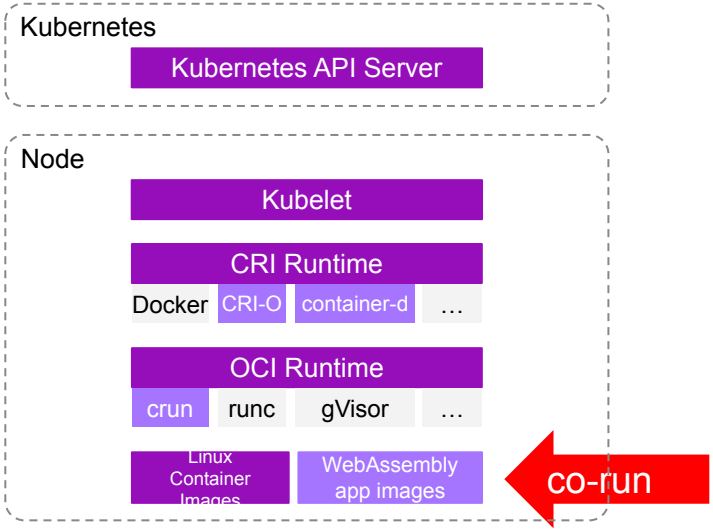


A Krustlet Kubernetes Stack

Example implementations

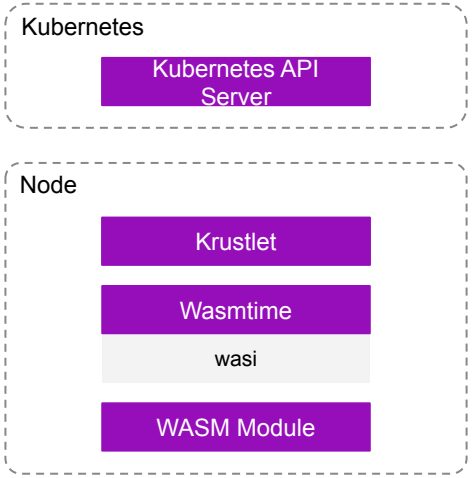


A Krustlet Kubernetes Stack

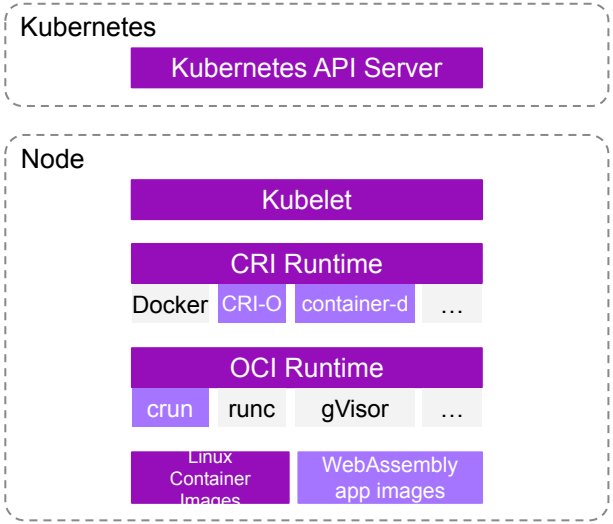


The Container Eco-System

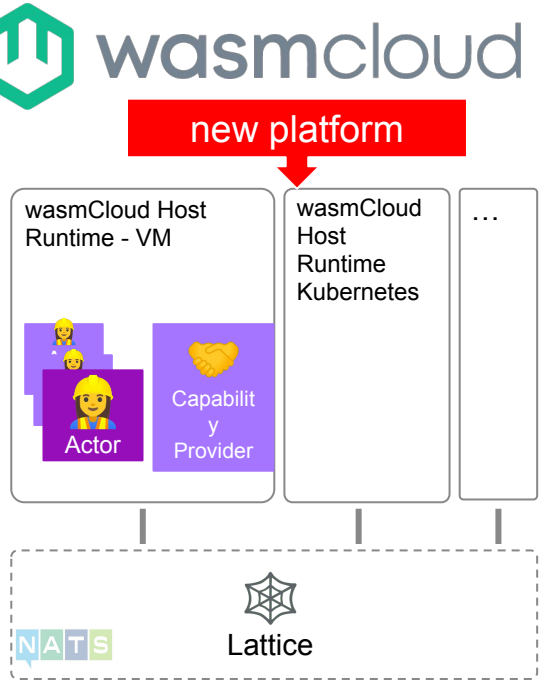
Example implementations



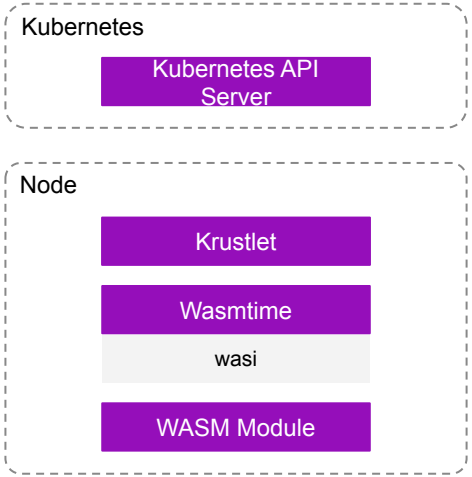
A Krustlet Kubernetes Stack



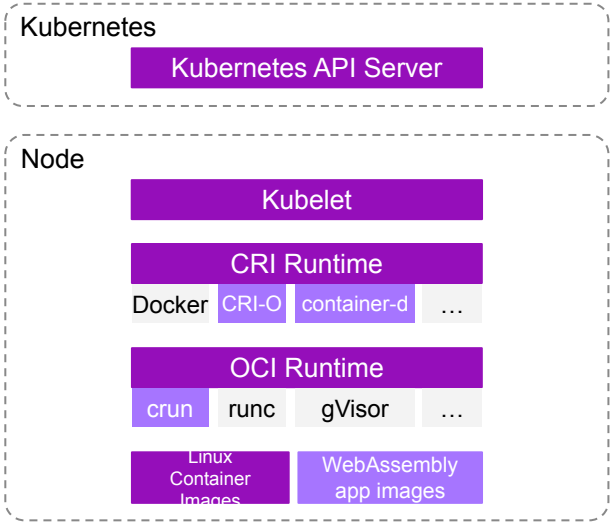
The Container Eco-System



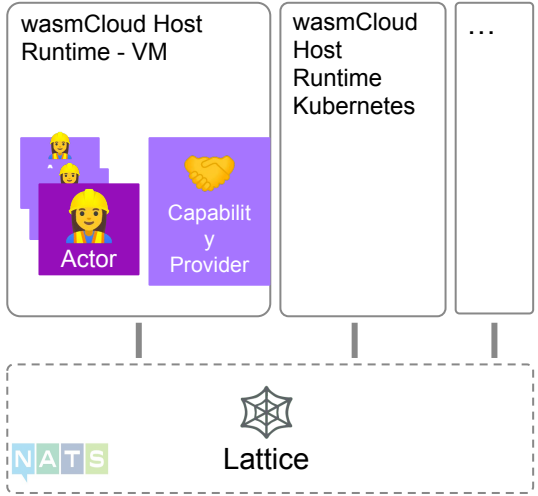
Example implementations



A Krustlet Kubernetes Stack



The Container Eco-System

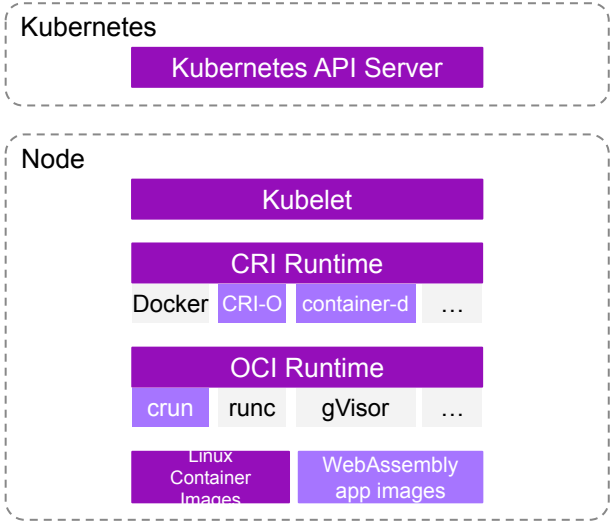


Let’s think about the WASM potentials based on WasmEdge

- Especially targets the integration in various Kubernetes distributions, CRI runtimes as well as OCI runtimes - therefore a good match to run WASM side by side with classic containers

- Runs also stand alone for modern web apps, to host serverless functions and being “embedded” in any kind of edge device.

- It leverages all advantages of WASM and bring it into a strong ecosystem without being inversive



The Container Eco-System

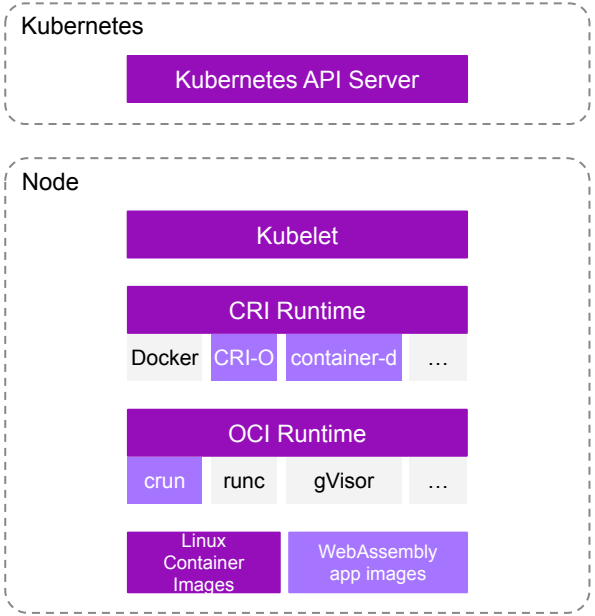
WasmEdge

Integrating with existing tooling, and more ...

- Especially targets the integration in various Kubernetes distributions, CRI runtimes as well as OCI runtimes - therefore a good match to run WASM side by side with classic containers

- Runs also stand alone for modern web apps, to host serverless functions and being “embedded” in any kind of edge device.

WasmEdgeRuntime



The Container Eco-System

based on: https://wasmedge.org/book/en/kubernetes.html

WasmEdge

Solution Approach

WasmEdge is different on the image level. Rather than having a container image with a OS, the WASM image is build from scratch. In addition, the container requires a “wasm.image” annotation, to let crun and containerd know that it use WasmEdge.

This approach allows to use WASM within the Kubernetes context, and utilize the existing ecosystem.

```
FROM scratch
ADD http_server.wasm /
CMD ["/http_server.wasm"]
```

*http server wasm image within a docker file

```
sudo buildah build --annotation "module.wasm.image/variant=compat" -t http_server .
```

*a wasm container requires the wasm image annotation

Demo

WasmEdge

Solution Approach

Advantages

- ➕ WasmEdge can run alongside your standard containers
- ➕ Build and deployment spec are nearly the same as for a normal pod
- ➕ Supports different CRI, OCI and K8s distros
- ➕ Can use existing K8s ecosystem
- ➕ Runs by itself on edge, serverless or browser

Considerations

- ➖ Additional tools for image annotation are required (at the moment)
- ➖ For some use cases you need another SDK
- ➖ It can lead to confusion that you can use WasmEdge in very different scenarios and each of them has to be developed differently

WasmEdge would be the best choice to extend your currently orchestration without deep cutting changes

WASM can extend Container

	Docker-like container	WebAssembly
Performance	OK	Great
Resource footprint	Poor	Great
Isolation	OK	Great
Safety	OK	Great
Portability	OK	Great
Security	OK	Great
Language and framework choice	Great	OK (yet)
Ease of use	Great	OK (yet)
Manageability	Great	Great

1

WebAssembly's potential is beyond the browser

2

WASM enables use cases that are not possible with container & K8s

3

WASM will not substitute containers & K8s, but extend it

4

WASM lacks harmonization and makes it difficult for programming languages to adapt

5

The developer experience of/for WASM will be the game changer

Containers for lifting, WASM for re-creating

Go with the Container flow

Containers will stay and drastically increase in usage over the next years.

- 🧸 **Simple to use**
- 🍀 **Big eco-system**
- 📱 **Language support**
- 👶 **“1st born” effect**

Build with WASM for the future

But for future developments WASM might be in many cases a better choice.

- ⚡ **Consistently fast**
- 🔬 **Small**
- 🌍 **Universal**
- ♻️ **Reusable**

We believe that WASM & Container will go along side by side