

WebAssembly on Kubernetes

Nicolas Fränkel

Me, myself, and I

- Software Engineer
- Developer Advocate
- Back to Software Engineer

Disclaimer

- The talk's material is not part of my daily job
- I'm just interested in the subject
- Take everything I say in this talk with a grain of salt

I assume everyone knows Kubernetes

“Kubernetes, also known as K8s, is an open source system for automating deployment, scaling, and management of containerized applications.

-- <https://kubernetes.io/>”



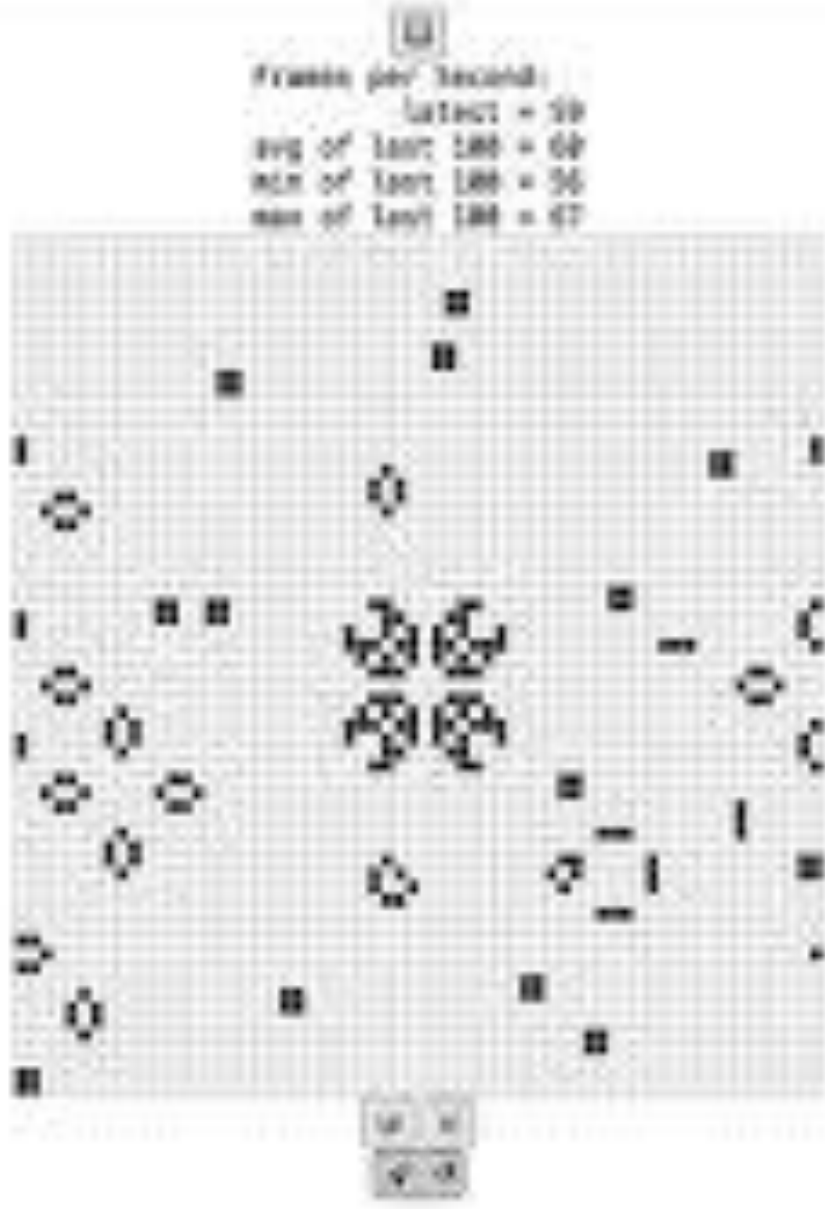
WebAssembly in 2018

“WebAssembly or wasm is a new portable, size- and load-time-efficient format suitable for **compilation to the web**.

-- <https://webassembly.org/>”

Use-cases

- Performance-critical code in browsers
 - Multimedia
 - 3D
 - Games



WebAssembly in 2020

“WebAssembly (abbreviated Wasm) is a **binary instruction format for a stack-based virtual machine**. Wasm is designed as a portable target for compilation of high-level languages like C/C++/Rust, enabling **deployment on the web for client and server applications**.

-- <https://webassembly.org/>”

Use-cases

- Polyglot approach
- Plugin architecture
- Function-as-a-Service
- Run at the edge on “small” devices

Write Once, Run Anywhere?



WebAssembly now

“WebAssembly (abbreviated Wasm) is a binary instruction format for a stack-based virtual machine. Wasm is designed as a portable compilation target **for programming languages**, enabling deployment on the web for client and server applications.

-- <https://webassembly.org/>”

Python can Wasm too!

“Python on the Edge: Fast, sandboxed, and powered by WebAssembly
We are excited to announce full Python support in Wasmer Edge (Beta)

-- <https://wasmer.io/posts/python-on-the-edge-powered-by-webassembly>”

Use-cases

- Everything
- Everywhere
- All at once

WebAssembly Is Now About More Than the Web

- Strong security
- Small binary sizes
- Fast loading and running
- Support for many operating systems and architectures
- Interoperability with the browser

-- <https://www.fermyon.com/blog/how-to-think-about-wasm>

The many moving facets of Wasm on Kubernetes

- [Krustlet](#): “Run WebAssembly workloads in your Kubernetes cluster.”
- [runwasi](#): “facilitate running wasm workloads managed by containerd either directly (ie. through ctr) or as directed by Kubelet via the CRI plugin”
- [SpinKube](#): “streamlines developing, deploying, and operating WebAssembly (Wasm) workloads in Kubernetes”
- [wasmCloud](#): “enables teams to build polyglot applications composed of reusable Wasm components and run them [...] across any cloud, Kubernetes, datacenter, or edge.”

The many moving facets of Wasm on Kubernetes

- [WasmEdge](#): “lightweight, high-performance , and extensible WebAssembly runtime for cloud-native, edge, and decentralized applications”
- [Wasmer](#): “allows running WebAssembly modules and Wasmer packages anywhere. Wasmer can be used as a library from any programming language, or as a standalone runtime”
- [Envoy Proxy wasm extensions](#): “extend the functionality of Envoy Gateway by running custom code against HTTP requests and responses, without modifying the Envoy Gateway binary”
- ...

My demo points today

- Container size
- Build chain (from Rust)
- Kubernetes-side configuration

Project Architecture

- Run containers in Kubernetes
- HTTP server based on Rust
 - Axum + serde
- Three different flavors
 - Compiled to native **FROM** distroless/static
 - Compiled to wasm **FROM** a wasm runtime
 - Compiled to wasm **FROM** scratch using a *containerd* wasm runtime

Project structure

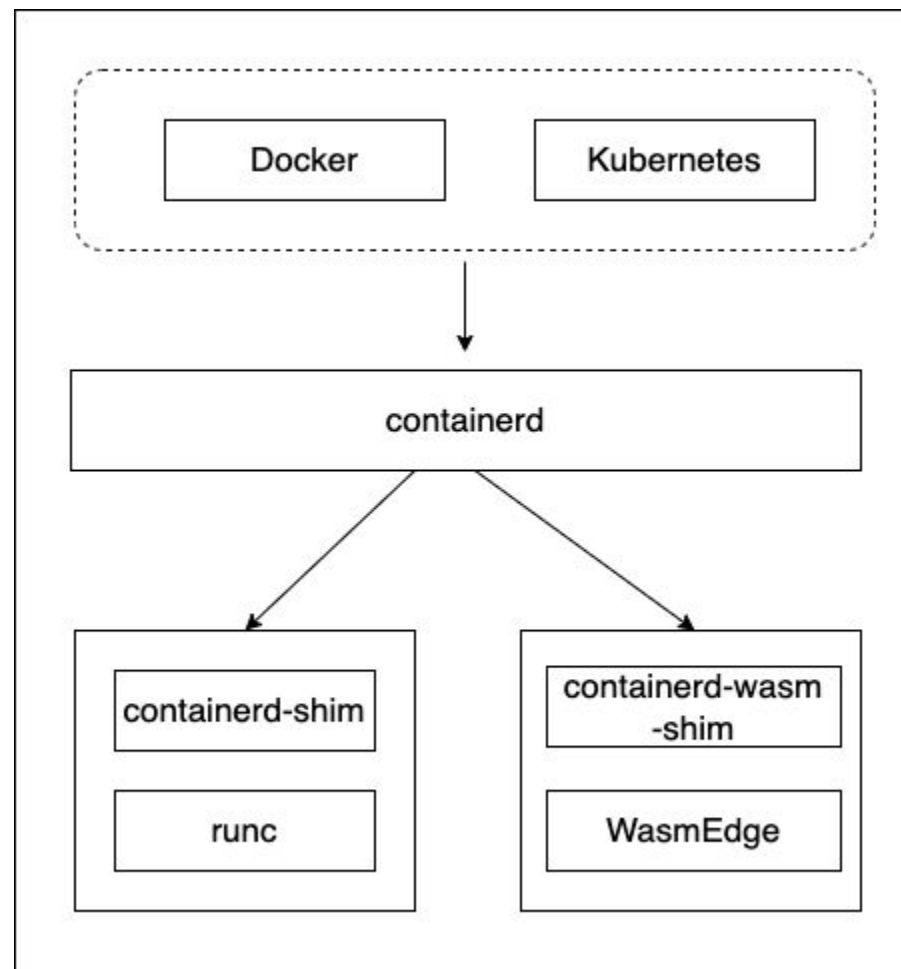
- One single code base
 - Emulates httpbin
 - Just for GET requests
- Two Cargo TOML
 - Native
 - Wasm
- Three Dockerfiles

A look at the code



<https://github.com/ajavageek/wasm-kubernetes>

Just enough Kubernetes intrinsics



Layer	Role	Responsibilities
containerd	Container lifecycle manager	● Pulling, storing, and managing images ● Handling container lifecycle (create, start, stop, delete)
runc	Low-level container runtime	● Creates containers from an OCI bundle ● Sets up namespaces, cgroups, seccomp, capabilities ● CLI tool
shim	Middleman	● Keep the container process attached to containerd even though runc has exited ● Forward stdin/stdout/stderr from the container to the higher layers ● Report container status back to containerd ● Ensure that if containerd restarts, the containers keep running

My choice: wasmedge

- I liked the implementation architecture
- Spoiler: it might not have been the best choice



The good

- An HTTP server in Wasm!
- Smallest image size
- Few changes to Kubernetes

The bad

- Code isn't actively maintained
 - Last commit of [wasi_tokio](#): March 30th, 2024
 - Last releases of [wasmedge/slim-runtime](#):
 - 0.13.5: almost 2 years ago
 - 0.14.0-alpha.1: 1 year ago

The ugly

- Code from 6 months ago didn't work anymore
 - I had to downgrade axum version from 0.8 to 0.6
- I still don't understand why it worked
 - Toolchain issue?
 - Bugfix version broke something?

Summary

- The WebAssembly/Kubernetes world is evolving fast
- If you're a startup living on the bleeding edge, enjoy
- If not, it might not be your best option

Thanks for your attention

- nfrankel.bsky.social 
- @nico@frankel.ch 
- @nicolas_frankel 
- <https://github.com/ajavageek/wasm-kubernetes>