WasmCon Workshop

From Sketch to Scale

Building and Scaling Wasm Components

cosmonic

Any Cloud, Any Edge, at Any Scale

What's on the Agenda

- Introductions
- Wasm? WASI? Component Model?
- What is wasmCloud?
- What is Cosmonic?
- Building with the `wit` IDL
- Deploying and scaling Wasm components (actors)
- Working with contracts

Introductions

Wasm Future

- Bytecode Alliance
  https://github.com/bytecodealliance
- Component Model
  https://github.com/WebAssembly/component-model
- WASI
  https://github.com/WebAssembly/WASI

- Standards Focused, CNCF Open Source Project
- Built on Wasmtime (rust, memory-safe)
- Secure out of the box (blog)
- WASI enabled
- Component Model (blog)
- Write “just” business logic
- Contract-based, hot-swappable capabilities
- Distributed on any device

- Hosted wasmCloud
- Free Tier
- Any Cloud, Any Edge, even your own
- Production-grade Managed Capabilities (included)
  - HTTP Server
  - KeyValue Store
  - NATS Messaging

What is it good for?

Good

- Distributed data locations
- Heterogeneous environments
- Network-constrained apps
- Multi-region or cloud
- Failover

Better Elsewhere

- Single component applications
- Web Front-ends
- CPU or OS-specific code
- Heavily optimized code

Let's Build Stuff

https://github.com/cosmonic/wasmcon-workshop

Start in Codespaces (recommended)

Select branch 0-start-here

Click Code > Create codespace

OR

Clone repo and download dev container

Clone repo

Open in VSCode, Install Dev Containers Extension

Reopen in Dev Container

Building an Actor Component

Use WASI interface to handle HTTP Request

Build and Deploy on Cosmonic

Adding KeyValue Interface

Use wasi:keyvalue to store/update records

Build and Deploy on Cosmonic

Hot-swap Capabilities

Change provider to Redis

✨ See the Magic ✨