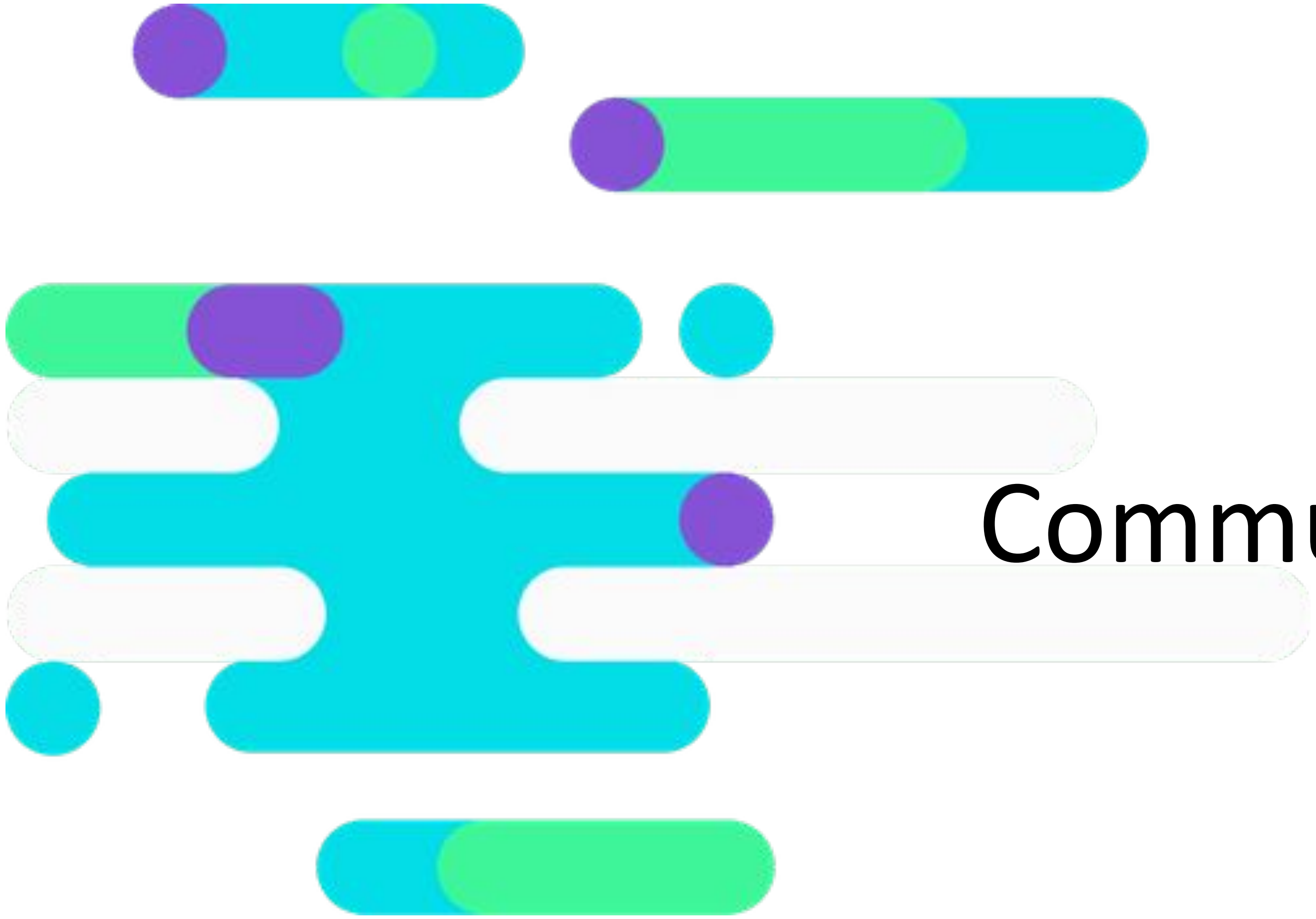
BYTECODE ALLIANCE



Community Meeting

January 31 2023

Agenda

- **Bytecode Alliance Community Meeting**
- **Bytecode Alliance Governance**
  - TSC
- **Projects**
  - wasmtime
  - Wasm-micro-runtime
- **WASI**
- **Component model**

BYTECODE ALLIANCE

Monthly Community Stream

- Bytecode Alliance Community Meeting
- Demos
- Celebrate recognized contributions

Questions?

#community-stream

BYTECODE ALLIANCE

Mission

Our mission is to provide state-of-the-art foundations to develop runtime environments and language toolchains where security, efficiency, and modularity can all coexist across a wide range of devices and architectures. We enable innovation in compilers, runtimes, and tooling, focusing on fine-grained sandboxing, capabilities-based security, modularity, and standards such as WebAssembly and WASI.

BYTECODE ALLIANCE

Governance

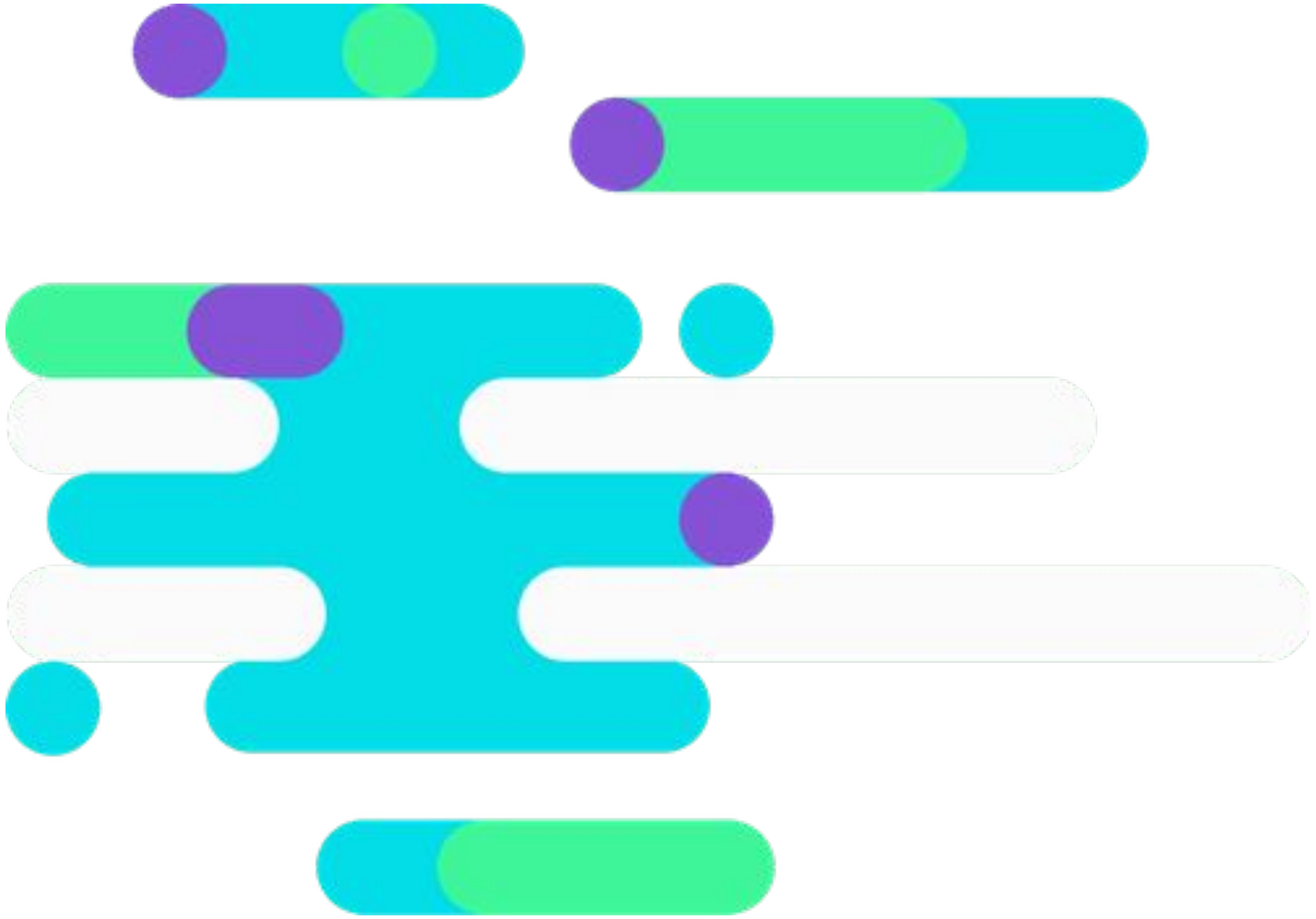
TSC Chair - Nick Fitzgerald

At-large-director - Till Schneidereit

Director - Bailey Hayes

BYTECODE ALLIANCE

BYTECODE ALLIANCE



WAMR

Xin Wang

Agenda

- WAMR overview
- 2022 achievements
- 2023 roadmap

BYTECODE ALLIANCE

WAMR overview

C implementation, small footprint, support interpreting, JIT and AoT compiling mode

Created by Intel, and donated to Bytecode Alliance in 2019

WAMR TSC by Intel, Amazon, Sony/Midokura, Alibaba/Ant, and Xiaomi members

Broad usages of WAMR

Trusted FaaS with SGX: Inclavare container, Apache Teaclavare, Faasm

Cloud/Edge: Carnegie Mellon University Silverline Cloud-Edge platform

Big Data: user-defined DB function

Devices: Amazon, Disney, Siemens, Alibaba..

Blockchain: Private Data Object, AntChain

Service mesh: Envoy proxy

IA: Sony

Embedded/IoT: Xiaomi IoT

BYTECODE ALLIANCE

WAMR community 2022 achievements

Fast JIT

Fast JIT to LLVM JIT tier up

GC early proposal on the interpreter (broken now)

Enhanced source debugger & VSCode extension

Bindings for Python and Go

SGX remote attestation, IPFS (Intel protected file system)

Socket API, wasi-nn, wasi-thread (WIP)

Nuttx system support for x86, riscv, xtensa, arm

Improved AoT solution for esp32

Envoy proxy new architecture support

BYTECODE ALLIANCE

The 2023 roadmap

Continue the performance improvement

GC latest proposal on interpreter and JIT

Support exception feature

Fast JIT SIMD support

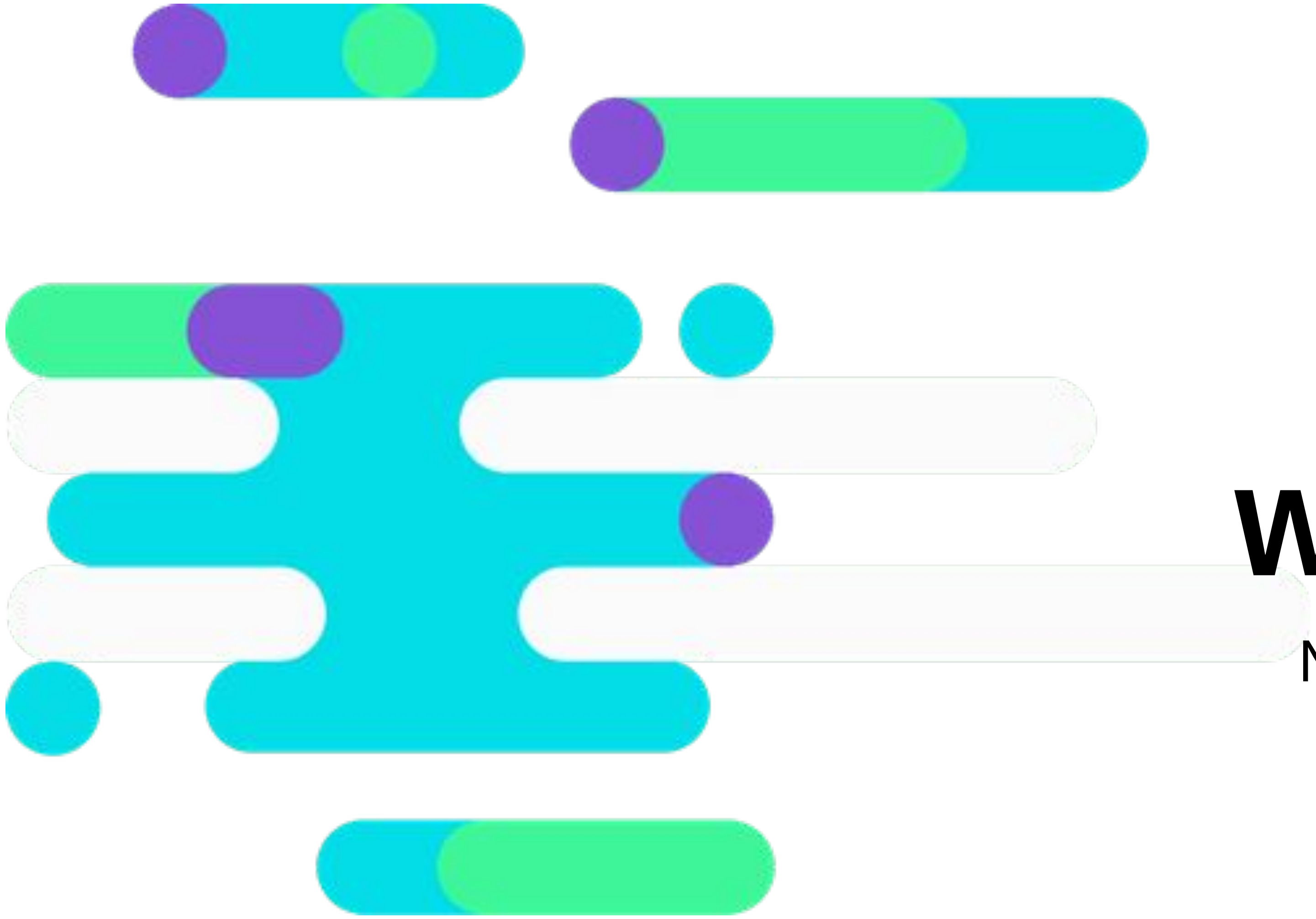
TypeScript to WebAssembly PoC

Component model (TBD)

Debugging support for multi-threading

BYTECODE ALLIANCE

BYTECODE ALLIANCE



Wasmtime

Nick Fitzgerald

Wasmtime 5.0 Released!

- Release Process
  - Major version every month
  - Security and correctness fixes backported to the most recent releases
- `wasmtime::component::bindgen!` macro

E-Graphs Mid-End Enabled by Default

- On by default on `main` branch
- Will let us run Wasm faster!
- Will let us verify correctness!

BYTECODE ALLIANCE

WASI Preview 2

Dan Gohman

WASI

- **The “WebAssembly System Interface”**
- **A Subgroup of the W3C WebAssembly Community Group**
- **Extend WebAssembly’s strengths to APIs:**
  - Portability
  - Security
  - Multi-language
  - Virtualizability

BYTECODE ALLIANCE

WASI Today: Preview 1

- **Preview 1 in action!**
  - Used in production
  - Lots of languages and engines
  - Launched the Subgroup
- **But it has limitations**
  - C-oriented
  - Sockets not well supported
  - Virtualization is difficult
  - Composition is difficult
- **Preview 1 taught us a lot**
  - Lessons learned incorporated into the component model and...

BYTECODE ALLIANCE

In development: WASI Preview 2

- **Built on component-model tooling**
  - Richer language for describing APIs
  - Idiomatic bindings in many source languages
- **Sockets**
  - including `listen` and `connect`
- **Leverages component-model strengths**
  - Virtualizable
  - Composable
  - Worlds
- **Go beyond POSIX**
  - Wasi-keyvalue, Wasi-messaging, Wasi-http-proxy, and more!

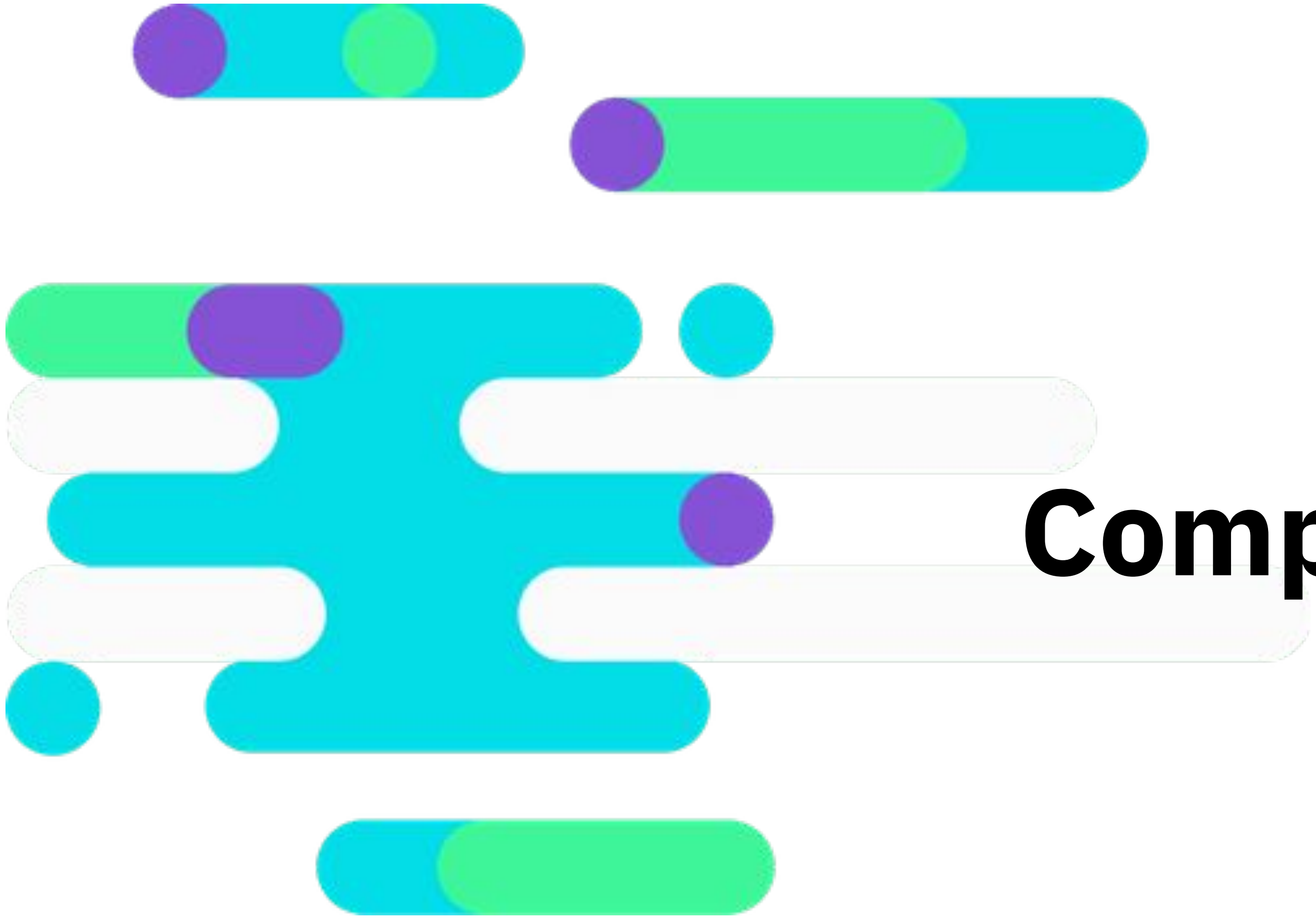
BYTECODE ALLIANCE

WASI Preview 2 Activities

- **Building:**
  - A Preview1-to-Preview2 converter
  - A Preview2 host implementation
  - Some language toolchains are starting to experiment with Preview2
  - I’m building demos for my upcoming Wasm I/O talk!
- **Stabilizing APIs**
- **Setting the stage for Preview 3 in the future**

BYTECODE ALLIANCE

BYTECODE ALLIANCE



Component Model

Luke Wagner

Four big-ticket features (from The Path to Components)

	Wasmtime implementation	Producer toolchains
parametric linking	✅	wit-component wasm-compose cargo component ...
high-level value types	✅	wit-bindgen
resource and handle types		
future and stream types		

wit-bindgen {

Bindings Generators (in progress)

Guest:

`Rust, C, Java (TeaVM), Go (TinyGo), JS (SpiderMonkey)`

Host:

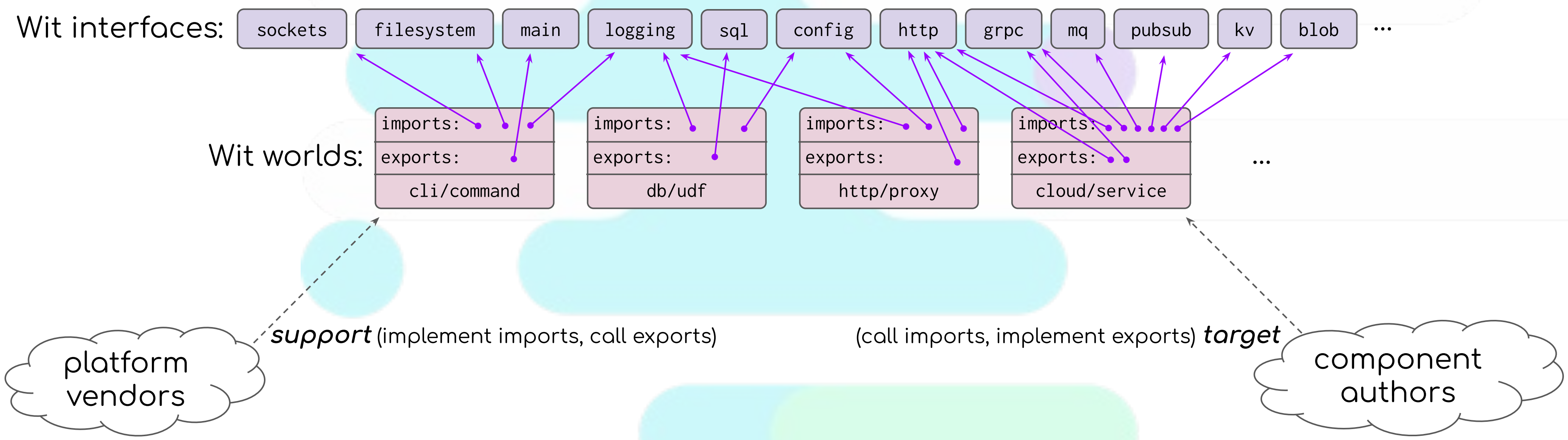
`Rust (Wasmtime), Python (wasmtime-py), JS (js-component-tools)`

→ browser/node polyfill!
polyfills itself to run in node!

BYTECODE ALLIANCE

Progress on Wit and “Worlds”

Recap (also from The Path to Components):



Progress on Wit and “Worlds”

- **Lots of work to make this real**
  - Compiling worlds to binary component types
  - Plumbing worlds through the whole Wit pipeline
  - Nicely resolving long-standing type-sharing issues
- **Created “Wit packages”**
  - A collection of interfaces and/or worlds
  - Can have dependencies on other Wit packages
  - Ideal unit of standardization and sharing (via registry…)
  - Lets us finally develop WASI as *modular* set of interfaces
- **Implemented in the core tools**
  - Still propagating outward

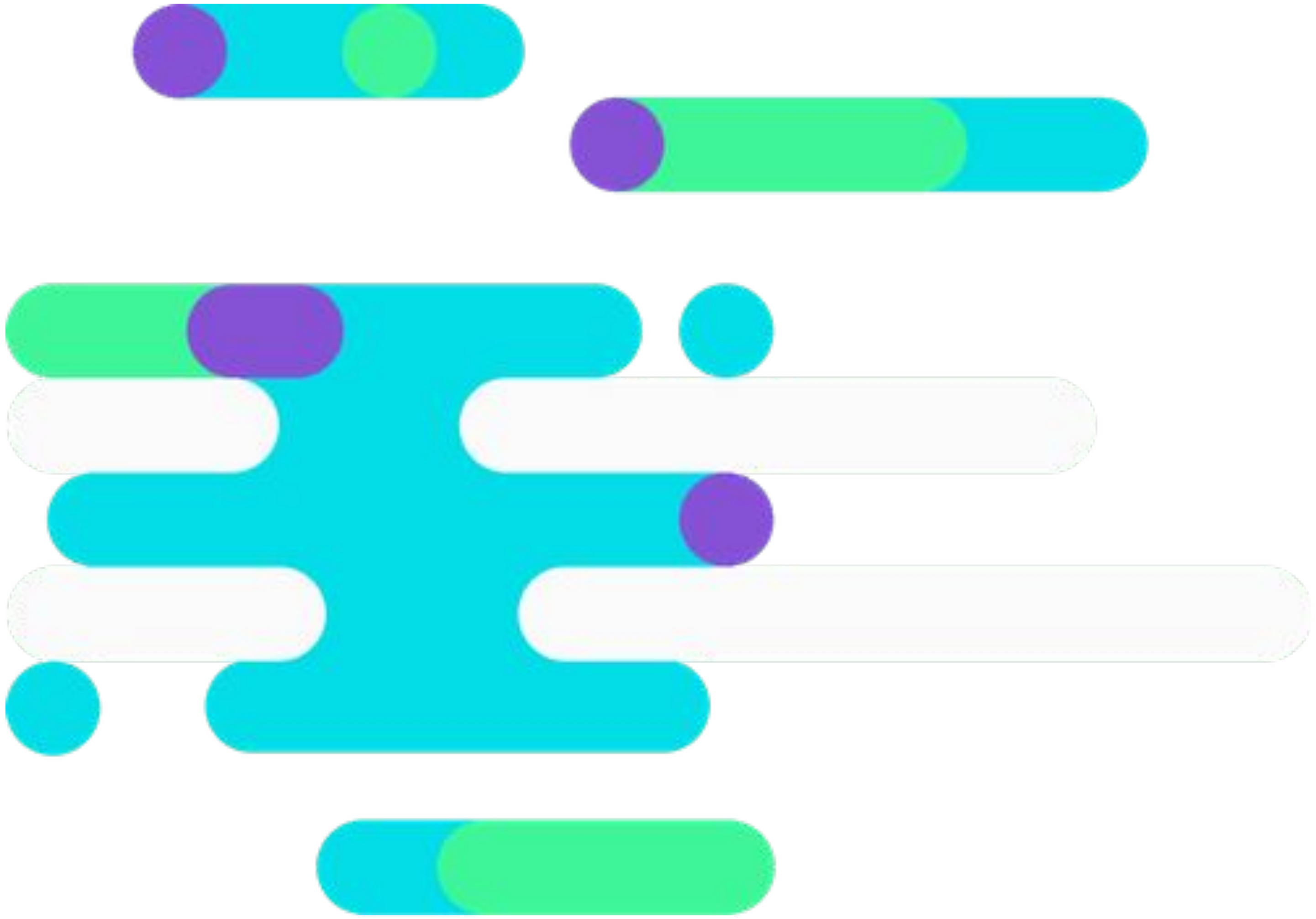
BYTECODE ALLIANCE

Next: resources and handles

- **Implementing in Wit, Wasmtime, producer tools**
  - Over the next few months
- **Then use in WASI Preview 2**
  - Fine-grained isolation of capabilities via handles
  - Virtualization (implement any interface with components)
  - Finally realize these original WASI goals
- **Check out the low-level design in the spec repo**
  - Assembly-level explainer
  - Wit explainer

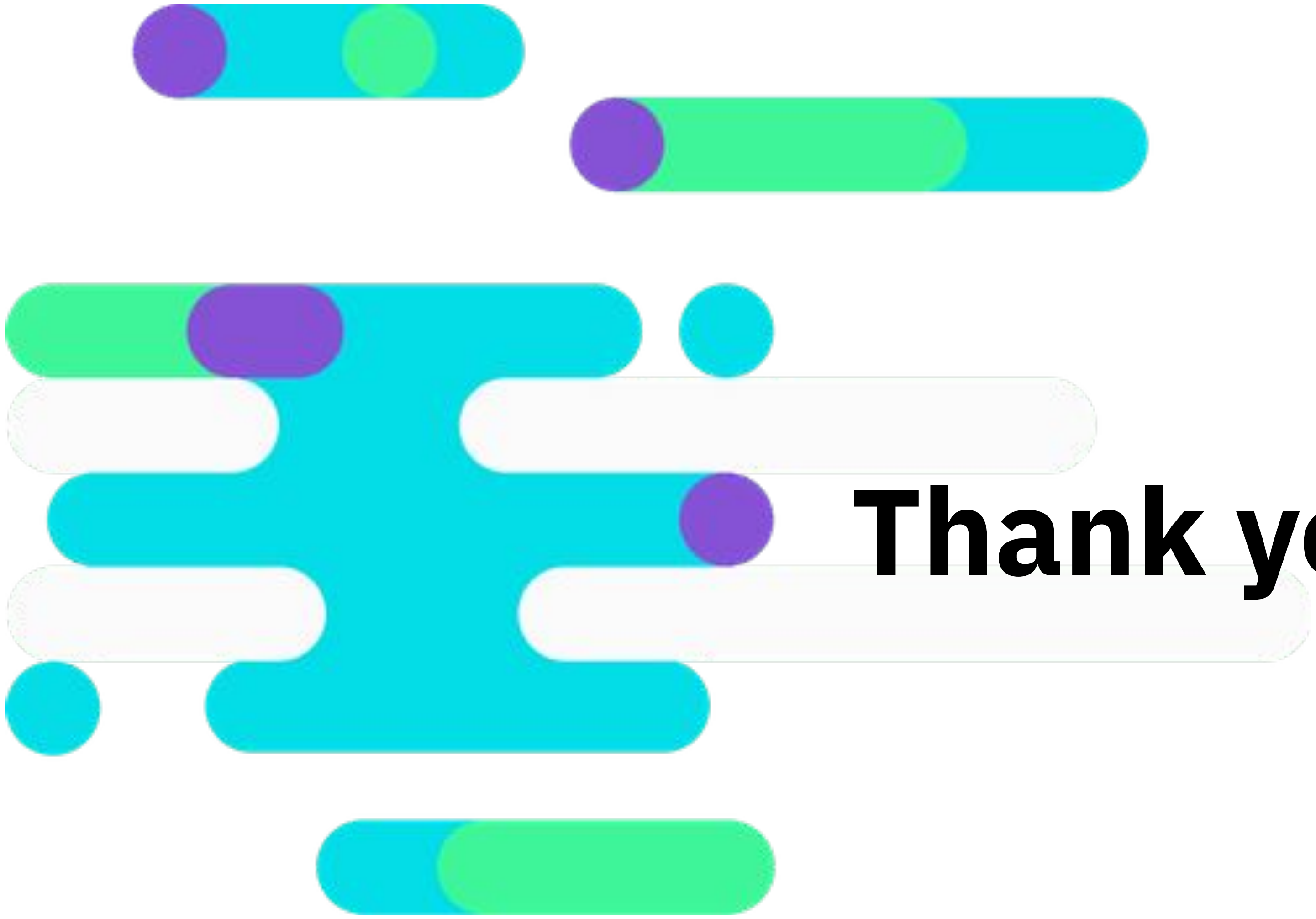
BYTECODE ALLIANCE

BYTECODE ALLIANCE



Q&A

BYTECODE ALLIANCE



Thank you for watching!