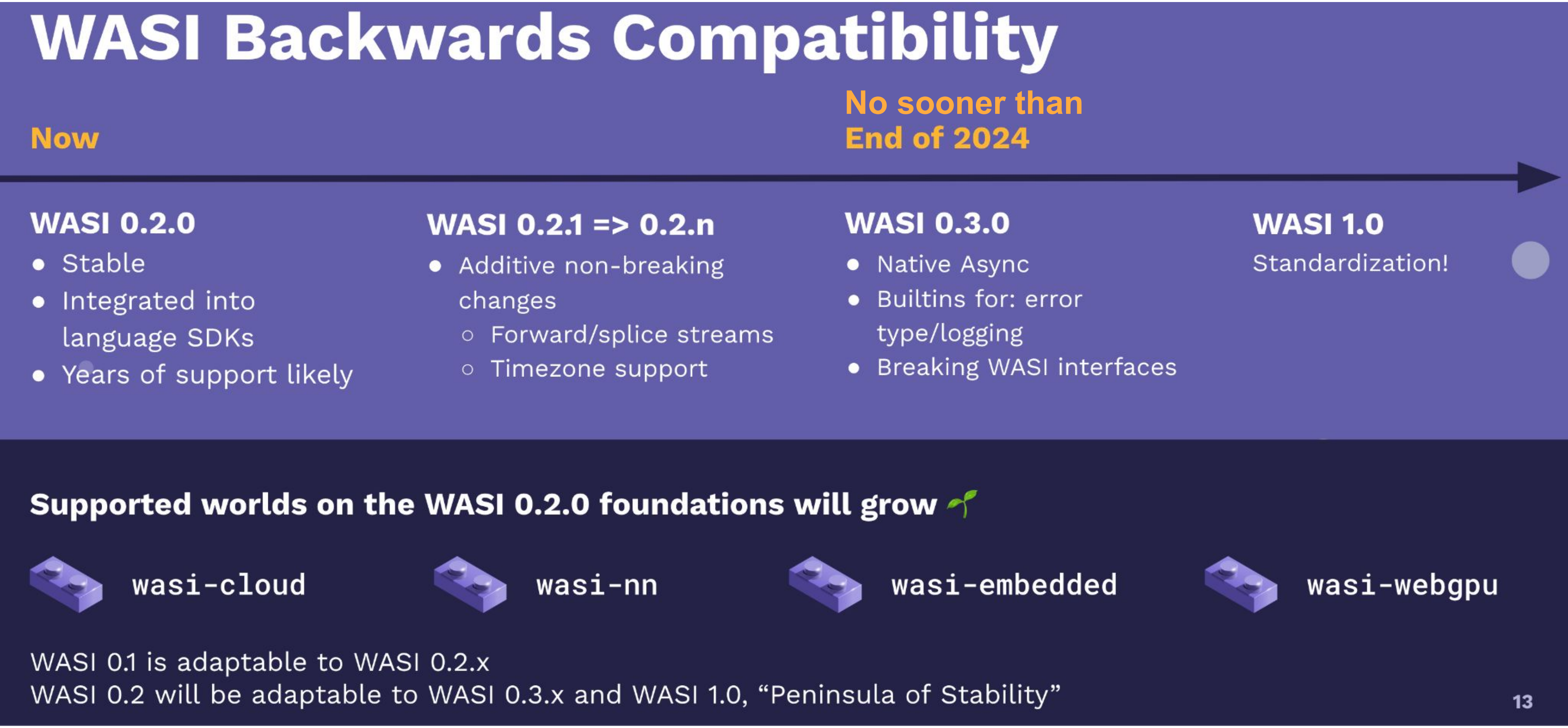
WASI 0.2.x Update

Pat Hickey and Luke Wagner



WASI Backwards Compatibility

Now

No sooner than End of 2024

WASI 0.2.0

- Stable
- Integrated into language SDKs
- Years of support likely

WASI 0.2.1 => 0.2.n

- Additive non-breaking changes
  - Forward/splice streams
  - Timezone support

WASI 0.3.0

- Native Async
- Builtins for: error type/logging
- Breaking WASI interfaces

WASI 1.0

Standardization!

Supported worlds on the WASI 0.2.0 foundations will grow 🌱

`wasi-cloud` `wasi-nn` `wasi-embedded` `wasi-webgpu`

WASI 0.1 is adaptable to WASI 0.2.x
WASI 0.2 will be adaptable to WASI 0.3.x and WASI 1.0, “Peninsula of Stability”

Proposals that are not yet part of Preview 2?

- 0.2.x, with no suffix, reserved for once subgroup has voted to include in Preview 2.

- 0.2.0-draft

  Early proposals have lots of churn, and aren’t stable enough for a version number.

- 0.2.0-rc-<iso 8601 date>

  Suggested for Phase 3.

Ultimately, up to proposal champions how to use these.

Proposals that are part of Preview 2?

- 0.2.x, with no suffix, reserved for when subgroup votes to include packages in a 0.2.x point release.

- 0.2.x-draft

  Work in progress.

- 0.2.x-rc-<iso 8601 date>

  Suggested as point release gets close.

Ultimately, up to proposal champions how to use these.

Point release roadmap

0.2.1:

- Only add timezone to wasi-clocks.
- Establish the cadence, intentionally small to ensure its easy.
- Make sure tooling can support point releases across ecosystem

0.2.2 or later, candidates:

- wasi-io: Stream forwarding
- wasi-http: https://github.com/WebAssembly/wasi-http/pull/103
- This list is incomplete, you can expand it by making PRs to proposals.

A Preview 2 by any other name

- What does Preview mean? Something specific to this group.
- But we are using Semver in the wit, and everyone already knows Semver.

So, call it:

- WASI 0.2 - recommended for most audiences. This is the version you'll see in wit documents and import names.
- WASIp2 or wasip2 - in a target triple
- WASI Preview 2 - WASI is not yet a formal Standard (requires WASM CG and WG), so in their context this is a Preview.

Target triples

- Go decided on wasm32-wasip1 and wasm32-wasip2
- Rust followed, introducing wasm32-wasip1 and wasm32-wasip2.

  https://github.com/rust-lang/rust/pull/120468: wasm32-wasip1 introduced, wasm32-wasi will remain accepted for some time with a warning to update, and eventually wasm32-wasi may be removed or changed to mean WASI 1.0.

  https://github.com/rust-lang/rust/pull/119616: wasm32-wasip2 introduced

- We suggest languages reserve wasm32-wasi if possible, to one day mean WASI 1.0.
- This subgroup can’t force these decisions, but language implementers may look to our group for guidance.