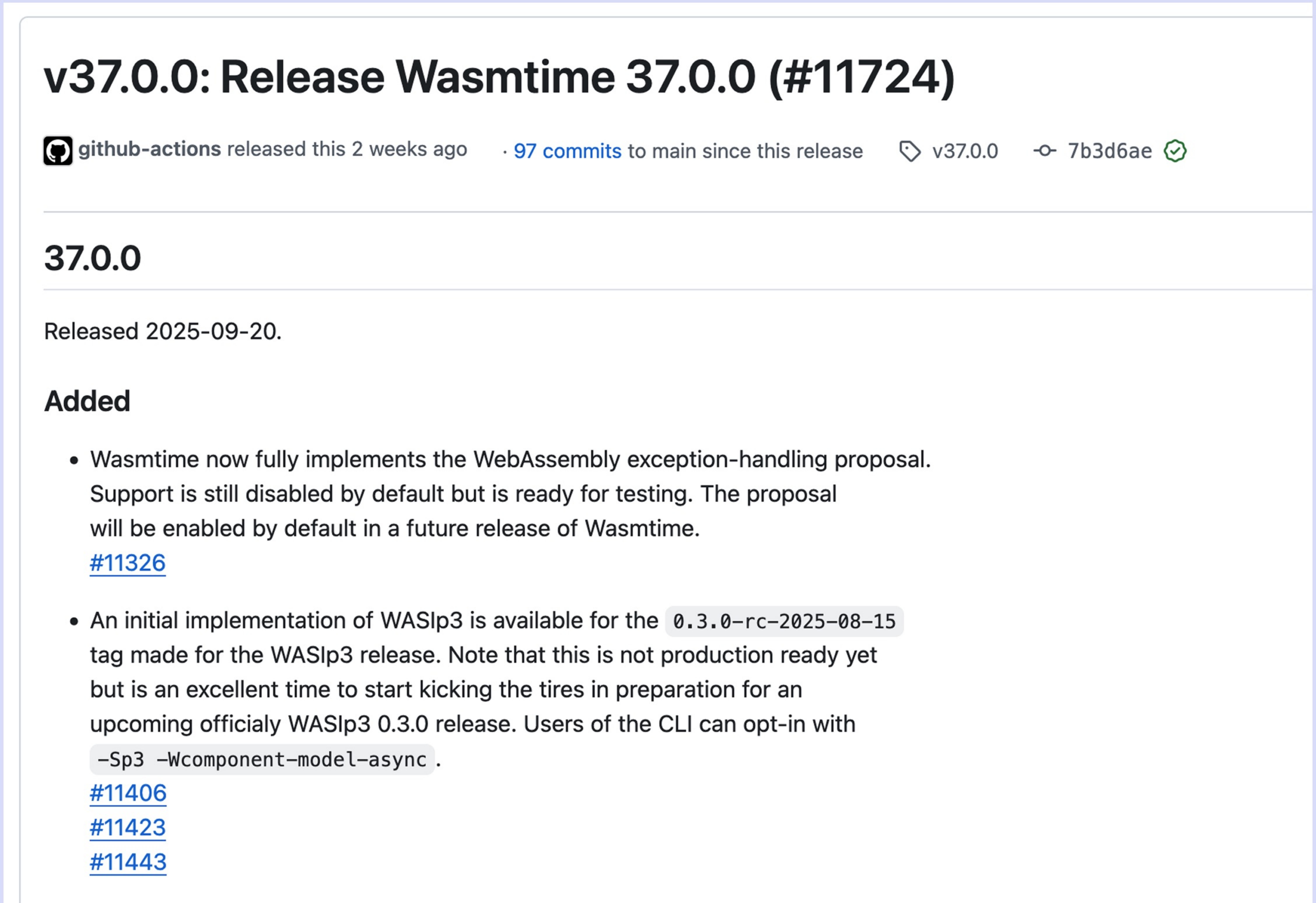
WASI 0.3.0 October Update

Yosh Wuyts
Microsoft

WASI SG Meeting
2025-10-02

Wasmtime



v37.0.0: Release Wasmtime 37.0.0 (#11724)

github-actions released this 2 weeks ago · 97 commits to main since this release v37.0.0 7b3d6ae

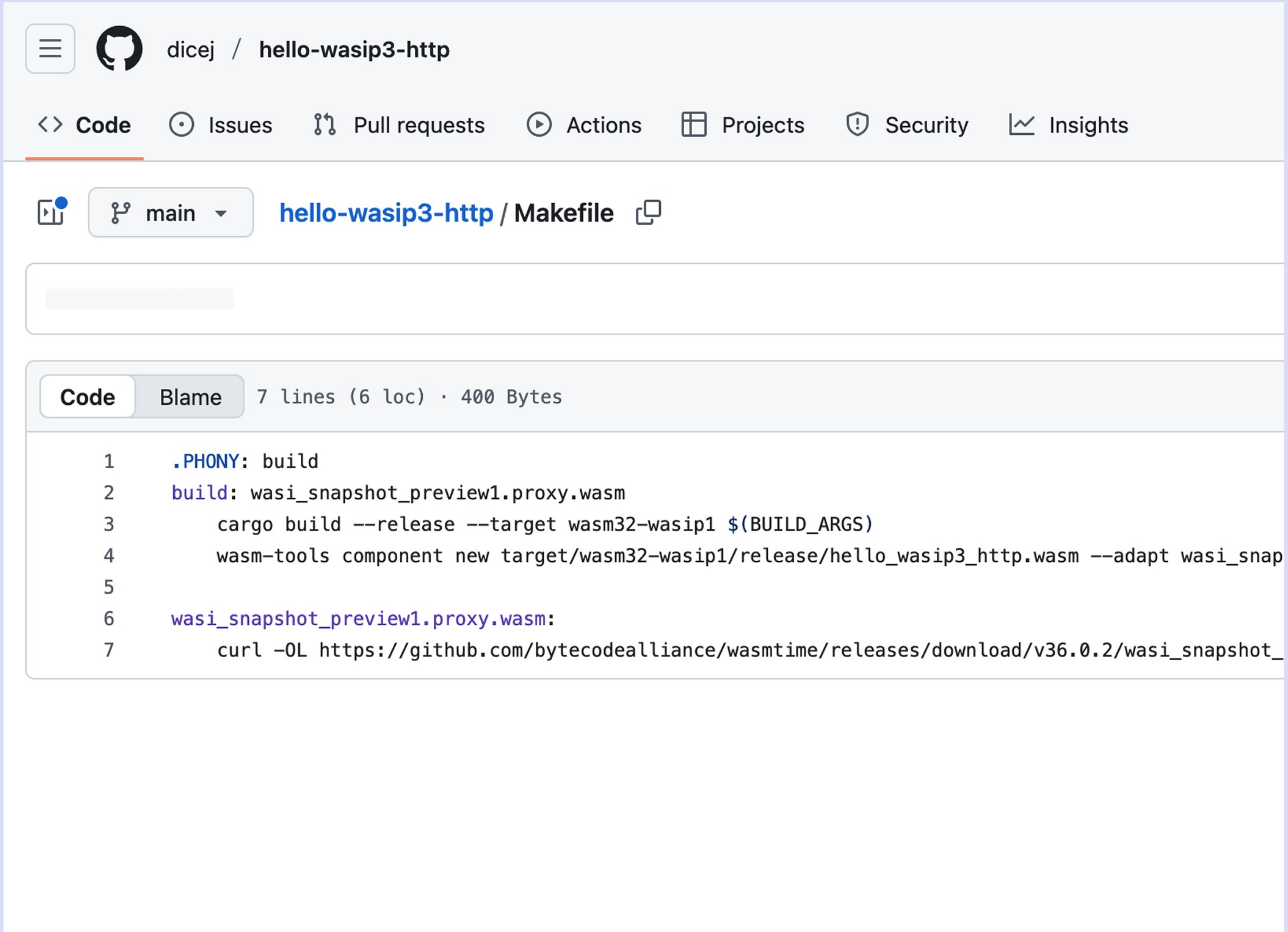
37.0.0

Released 2025-09-20.

Added

- Wasmtime now fully implements the WebAssembly exception-handling proposal. Support is still disabled by default but is ready for testing. The proposal will be enabled by default in a future release of Wasmtime.
  #11326

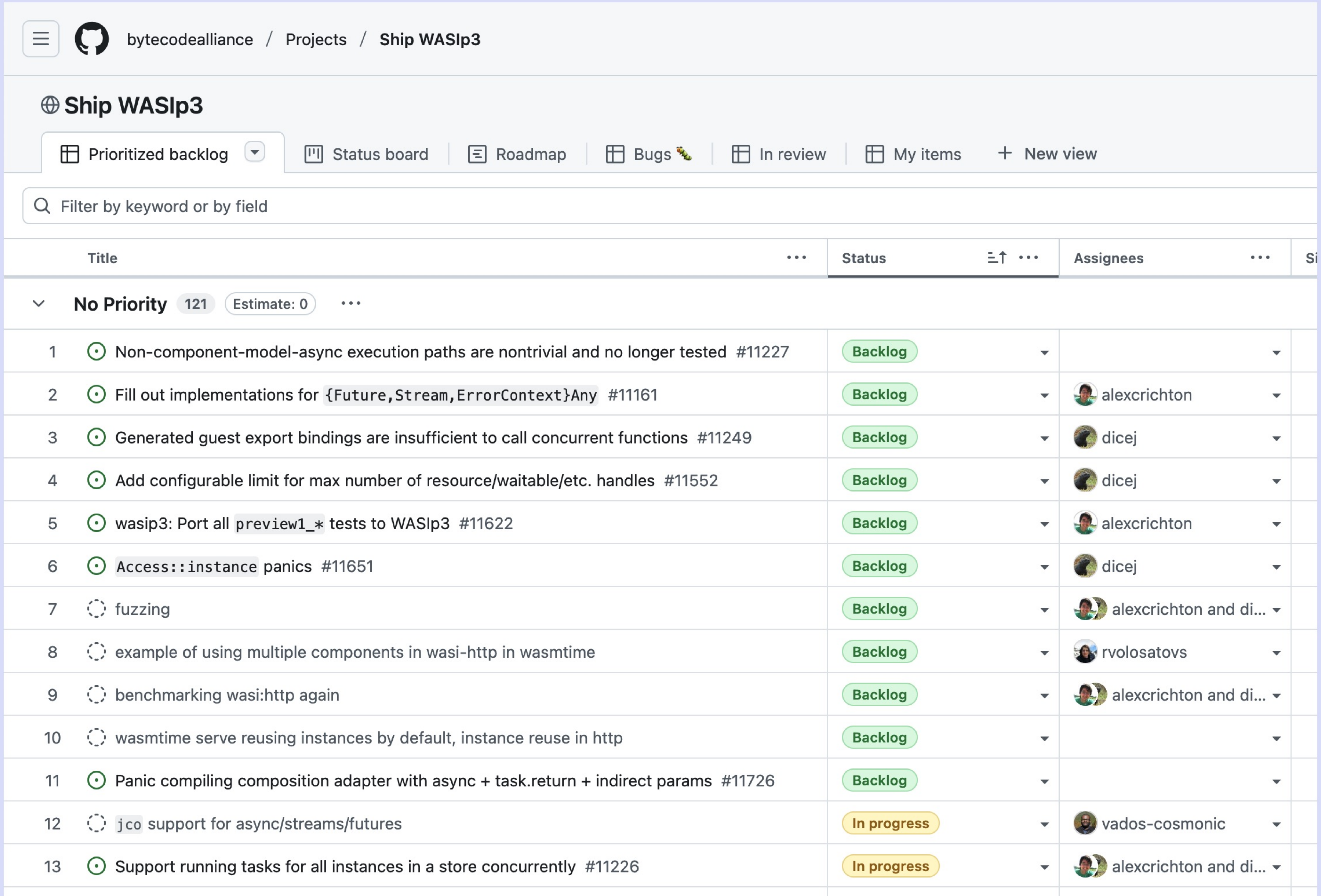
- An initial implementation of WASIp3 is available for the `0.3.0-rc-2025-08-15` tag made for the WASIp3 release. Note that this is not production ready yet but is an excellent time to start kicking the tires in preparation for an upcoming officialy WASIp3 0.3.0 release. Users of the CLI can opt-in with `-Sp3 -Wcomponent-model-async`.
  #11406
  #11423
  #11443



7 lines (6 loc) · 400 Bytes

```
.PHONY: build
build: wasi_snapshot_preview1.proxy.wasm
	cargo build --release --target wasm32-wasip1 $(BUILD_ARGS)
	wasm-tools component new target/wasm32-wasip1/release/hello_wasip3_http.wasm --adapt wasi_snap

wasi_snapshot_preview1.proxy.wasm:
	curl -OL https://github.com/bytecodealliance/wasmtime/releases/download/v36.0.2/wasi_snapshot_
```



bytecodealliance / Projects / Ship WASIp3

Ship WASIp3

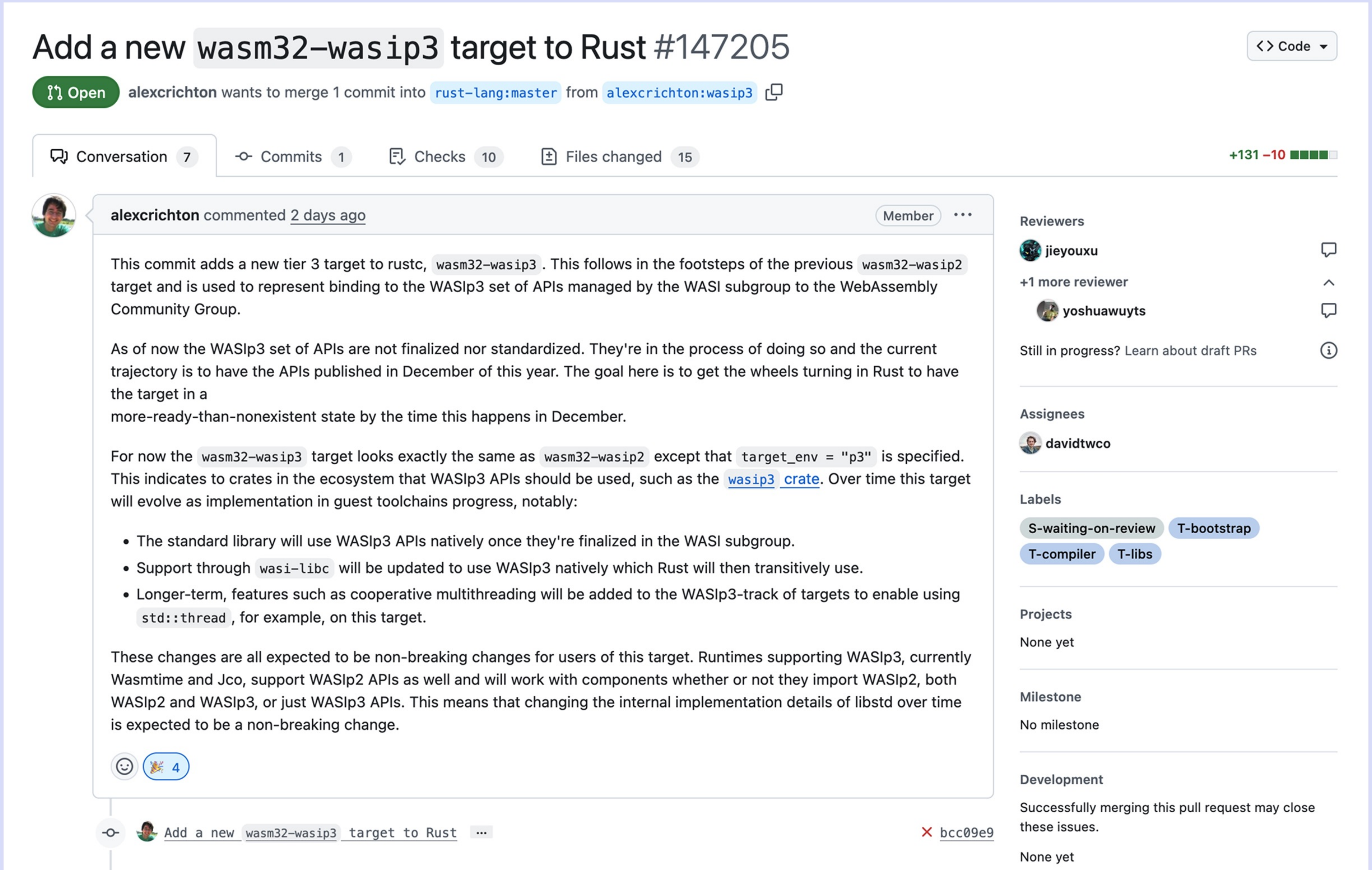
Prioritized backlog | Status board | Roadmap | Bugs 🐛 | In review | My items | + New view

Filter by keyword or by field

No Priority 121 Estimate: 0

	Title	Status	Assignees
1	Non-component-model-async execution paths are nontrivial and no longer tested #11227	Backlog	
2	Fill out implementations for `{Future,Stream,ErrorContext}Any` #11161	Backlog	alexcrichton
3	Generated guest export bindings are insufficient to call concurrent functions #11249	Backlog	dicej
4	Add configurable limit for max number of resource/waitable/etc. handles #11552	Backlog	dicej
5	wasip3: Port all `preview1_*` tests to WASIp3 #11622	Backlog	alexcrichton
6	`Access::instance` panics #11651	Backlog	dicej
7	fuzzing	Backlog	alexcrichton and di...
8	example of using multiple components in wasi-http in wasmtime	Backlog	rvolosatovs
9	benchmarking wasi:http again	Backlog	alexcrichton and di...
10	wasmtime serve reusing instances by default, instance reuse in http	Backlog	
11	Panic compiling composition adapter with async + task.return + indirect params #11726	Backlog	
12	`jco` support for async/streams/futures	In progress	vados-cosmonic
13	Support running tasks for all instances in a store concurrently #11226	In progress	alexcrichton and di...

Rust Toolchain



Add a new `wasm32-wasip3` target to Rust #147205

alexcrichton wants to merge 1 commit into `rust-lang:master` from `alexcrichton:wasip3`

alexcrichton commented 2 days ago

This commit adds a new tier 3 target to rustc, `wasm32-wasip3`. This follows in the footsteps of the previous `wasm32-wasip2` target and is used to represent binding to the WASIp3 set of APIs managed by the WASI subgroup to the WebAssembly Community Group.

As of now the WASIp3 set of APIs are not finalized nor standardized. They're in the process of doing so and the current trajectory is to have the APIs published in December of this year. The goal here is to get the wheels turning in Rust to have the target in a
more-ready-than-nonexistent state by the time this happens in December.

For now the `wasm32-wasip3` target looks exactly the same as `wasm32-wasip2` except that `target_env = "p3"` is specified. This indicates to crates in the ecosystem that WASIp3 APIs should be used, such as the `wasip3` crate. Over time this target will evolve as implementation in guest toolchains progress, notably:

- The standard library will use WASIp3 APIs natively once they're finalized in the WASI subgroup.
- Support through `wasi-libc` will be updated to use WASIp3 natively which Rust will then transitively use.
- Longer-term, features such as cooperative multithreading will be added to the WASIp3-track of targets to enable using `std::thread`, for example, on this target.

These changes are all expected to be non-breaking changes for users of this target. Runtimes supporting WASIp3, currently Wasmtime and Jco, support WASIp2 APIs as well and will work with components whether or not they import WASIp2, both WASIp2 and WASIp3, or just WASIp3 APIs. This means that changing the internal implementation details of libstd over time is expected to be a non-breaking change.

`Add a new wasm32-wasip3 target to Rust` … ✗ `bcc09e9`

Reviewers

jieyouxu

+1 more reviewer

yoshuawuyts

Still in progress? Learn about draft PRs

Assignees

davidtwco

Labels

S-waiting-on-review T-bootstrap T-compiler T-libs

Projects

None yet

Milestone

No milestone

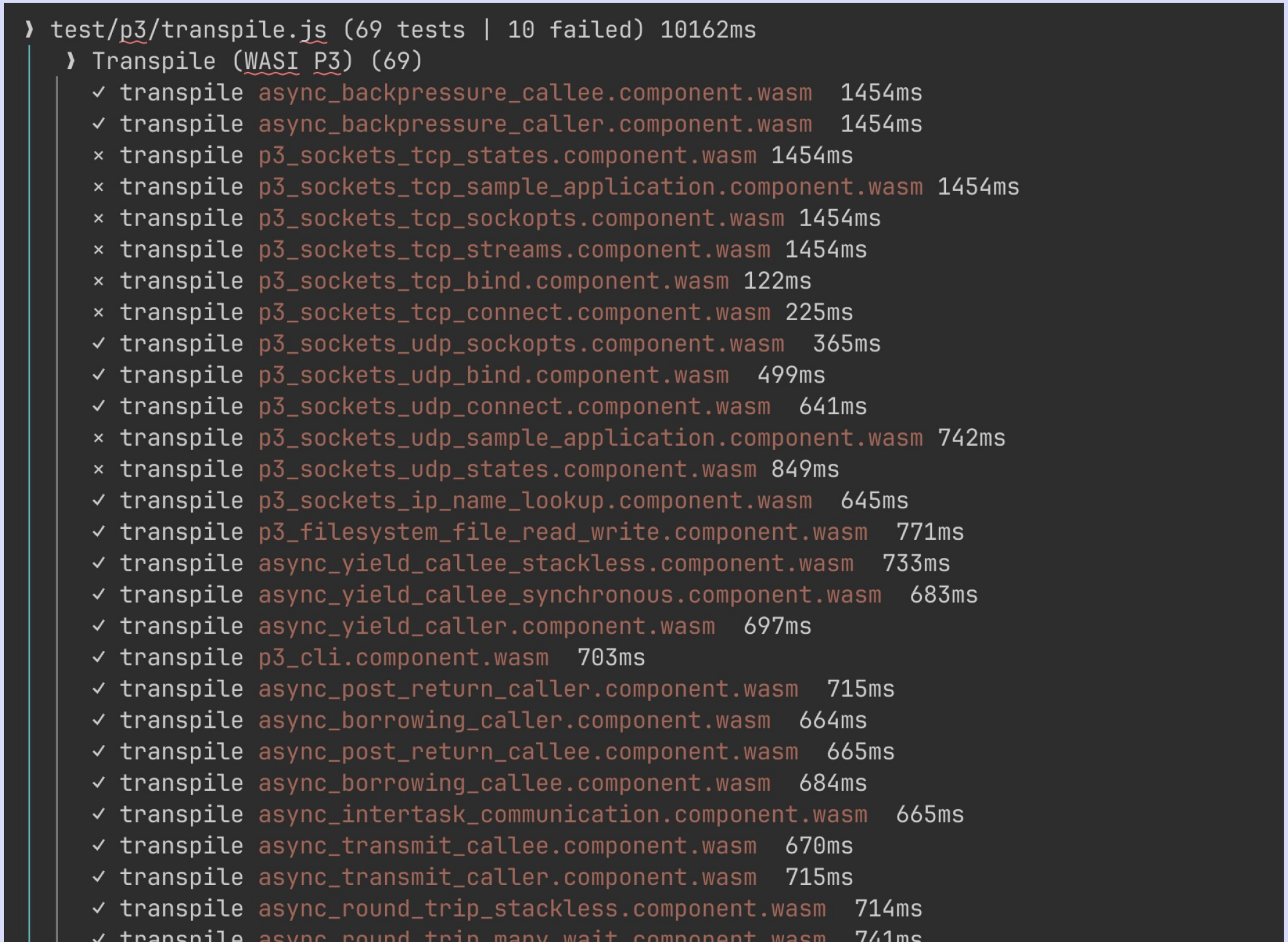
Development

Successfully merging this pull request may close these issues.

None yet

Jco

69/79 components passing



```
❯ test/p3/transpile.js (69 tests | 10 failed) 10162ms
   ❯ Transpile (WASI P3) (69)
     ✓ transpile async_backpressure_callee.component.wasm  1454ms
     ✓ transpile async_backpressure_caller.component.wasm  1454ms
     × transpile p3_sockets_tcp_states.component.wasm 1454ms
     × transpile p3_sockets_tcp_sample_application.component.wasm 1454ms
     × transpile p3_sockets_tcp_sockopts.component.wasm 1454ms
     × transpile p3_sockets_tcp_streams.component.wasm 1454ms
     × transpile p3_sockets_tcp_bind.component.wasm 122ms
     × transpile p3_sockets_tcp_connect.component.wasm 225ms
     ✓ transpile p3_sockets_udp_sockopts.component.wasm  365ms
     ✓ transpile p3_sockets_udp_bind.component.wasm  499ms
     ✓ transpile p3_sockets_udp_connect.component.wasm  641ms
     × transpile p3_sockets_udp_sample_application.component.wasm 742ms
     × transpile p3_sockets_udp_states.component.wasm 849ms
     ✓ transpile p3_sockets_ip_name_lookup.component.wasm  645ms
     ✓ transpile p3_filesystem_file_read_write.component.wasm  771ms
     ✓ transpile async_yield_callee_stackless.component.wasm  733ms
     ✓ transpile async_yield_callee_synchronous.component.wasm  683ms
     ✓ transpile async_yield_caller.component.wasm  697ms
     ✓ transpile p3_cli.component.wasm  703ms
     ✓ transpile async_post_return_caller.component.wasm  715ms
     ✓ transpile async_borrowing_caller.component.wasm  664ms
     ✓ transpile async_post_return_callee.component.wasm  665ms
     ✓ transpile async_borrowing_callee.component.wasm  684ms
     ✓ transpile async_intertask_communication.component.wasm  665ms
     ✓ transpile async_transmit_callee.component.wasm  670ms
     ✓ transpile async_transmit_caller.component.wasm  715ms
     ✓ transpile async_round_trip_stackless.component.wasm  714ms
     ✓ transpile async_round_trip_many_wait.component.wasm  741ms
```

Timeline

Timeline

- 2025-11-27 WASI SG Meeting
- 2025-12-04 WASI Release

Fin