WASIp3 Testing Status

16 Oct 2025—WASI CG

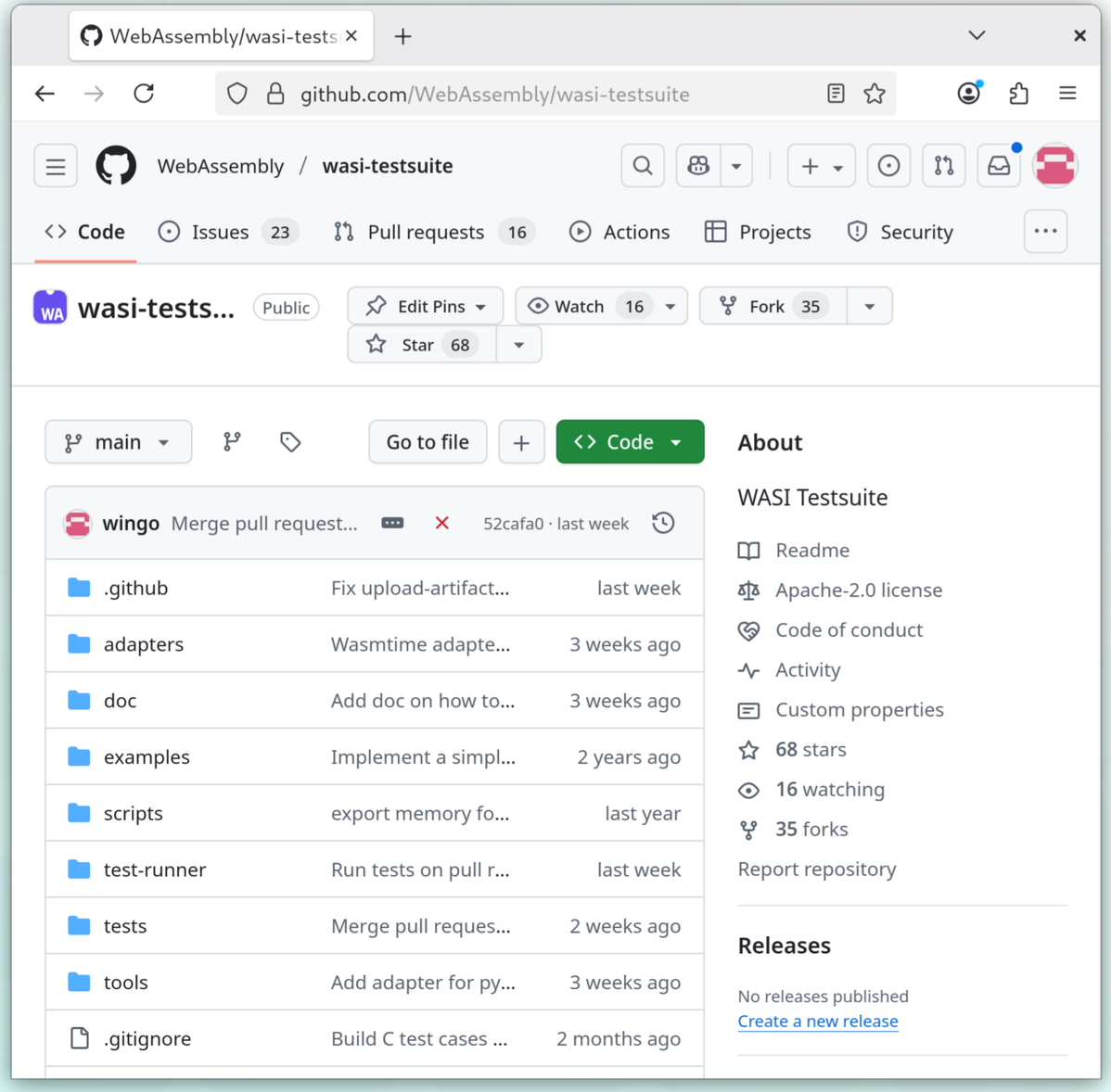
Andy Wingo

WASIp3 testing project

Goal: “Test WASIp3”

Sponsored by Bytecode Alliance; thanks!

wasi-testsuite



wasi-testsuite progress (1)

- Print command line when test fails
- Can test multiple WASI implementations per run
- Automatic enumeration of available WASI implementations, tests
- Concise output by default: `./run-tests`

wasi-testsuite progress (2)

More WASI implementations (“adapters”): wasmedge, pywasm, wazero, jco*

Different WASI versions; adapters declare which they support

Run CI on PRs: currently wasmtime on windows, ubuntu, mac

Builds on work by Marcin Kolny and others

```
[dev-env] wingo@beastie ~/src/wasip3/wasi-testsuite$ WASMTIME=~/src/wasip3/wasmtime/target/release/wasmtime ./run-tests
Detecting WASI runtime availability:
  wasmtime.py: wasmtime 38.0.0
  pywasm.py: pywasm 2.2.0
  wasm-micro-runtime.py unavailable; pass `--runtime /home/wingo/src/wasip3/wasi-testsuite/adapters/wasm-micro-runtime.py` to debug.
  wizard.py unavailable; pass `--runtime /home/wingo/src/wasip3/wasi-testsuite/adapters/wizard.py` to debug.
  wazero.py unavailable; pass `--runtime /home/wingo/src/wasip3/wasi-testsuite/adapters/wazero.py` to debug.
  wasmedge.py unavailable; pass `--runtime /home/wingo/src/wasip3/wasi-testsuite/adapters/wasmedge.py` to debug.

Running test suite WASI C tests [wasm32-wasip1] with wasmtime 38.0.0
..............
Running test suite WASI C tests [wasm32-wasip1] with pywasm 2.2.0
!.!.!.........


Running test suite WASI Rust tests [wasm32-wasip3] with wasmtime 38.0.0
..!..........!...!....!
Running test suite WASI Rust tests [wasm32-wasip3] with pywasm 2.2.0
_____________________
Running test suite WASI Rust tests [wasm32-wasip1] with wasmtime 38.0.0
................!............................
Running test suite WASI Rust tests [wasm32-wasip1] with pywasm 2.2.0
!!!!!!!.!!!!!.!.!!!!.!!!!!!!!!!!!!!!!!!!!!!!!
Running test suite WASI Assemblyscript  tests [wasm32-wasip1] with wasmtime 38.0.0
............
Running test suite WASI Assemblyscript  tests [wasm32-wasip1] with pywasm 2.2.0
!...!.......
===== Test results =====
wasmtime 38.0.0: FAIL: 5/95 tests failed
  /home/wingo/src/wasip3/wasmtime/target/release/wasmtime -Wcomponent-model-async -Sp3,http tests/rust/testsuite/wasm32-wasip3/test-stat-at-root.wasm
  /home/wingo/src/wasip3/wasmtime/target/release/wasmtime -Wcomponent-model-async -Sp3,http tests/rust/testsuite/wasm32-wasip3/field-invalid.wasm
  /home/wingo/src/wasip3/wasmtime/target/release/wasmtime -Wcomponent-model-async -Sp3,http tests/rust/testsuite/wasm32-wasip3/field-capitalization.wasm
  /home/wingo/src/wasip3/wasmtime/target/release/wasmtime -Wcomponent-model-async -Sp3,http tests/rust/testsuite/wasm32-wasip3/http-response.wasm
  /home/wingo/src/wasip3/wasmtime/target/release/wasmtime --dir tests/rust/testsuite/wasm32-wasip1/fs-tests.dir::fs-tests.dir tests/rust/testsuite/wasm32-was
pywasm 2.2.0: FAIL: 47/72 tests failed (23 skipped)
  /home/wingo/src/wasip3/wasi-testsuite/tools/run-pywasm --dir tests/c/testsuite/wasm32-wasip1/fs-tests.dir::fs-tests.dir tests/c/testsuite/wasm32-wasip1/pwr
  /home/wingo/src/wasip3/wasi-testsuite/tools/run-pywasm --dir tests/c/testsuite/wasm32-wasip1/fs-tests.dir::fs-tests.dir tests/c/testsuite/wasm32-wasip1/fdo
  /home/wingo/src/wasip3/wasi-testsuite/tools/run-pywasm --dir tests/c/testsuite/wasm32-wasip1/fs-tests.dir::fs-tests.dir tests/c/testsuite/wasm32-wasip1/pre
  /home/wingo/src/wasip3/wasi-testsuite/tools/run-pywasm --dir tests/rust/testsuite/wasm32-wasip1/fs-tests.dir::fs-tests.dir tests/rust/testsuite/wasm32-wasi
```

A wild goose chase (1)

WASIp1 underspecified, yet EOL as specification

- What errors are acceptable
- When can an implementation just `ENOTSUP`
- Rights/capabilities interface a dead-end
- Preopens weird

A wild goose chase (2)

Wasmtime failing a handful of tests… on Linux

…many, many more on Windows; wasi-testsuite CI does not do Windows

Alex: “wasi-testsuite just isn’t useful to Wasmtime right now”

WASIp3 inherits some of these WASIp1 problems, would be nice to solve

What to do?

False starts

Abortive attempt: let's just specify the one behavior

Problem: Windows and Linux admit different forms of file names

Problem: Different filesystems on same OS have different semantics

Dan: "`wasi-filesystem` is there to let users access their files"

This aspect of `wasi-filesystem` *should be documented*

Course correction

Are uniform semantics of primary importance to you? Use a database

`wasi-filesystem` should specify set of acceptable behavior, not the one behavior

Problems seem limited to `wasi-filesystem`

Alex: Pre-merge CI on Windows, Mac, Linux

Bugs remain in tests but also Wasmtime

Possible future directions

Formally deprecate subsystem. Encourage implementations to return `ENOTSUP` for WASIp1

Ditch preopens. WASI should provide sandboxed root FS instead

Explicitly reference POSIX. Specify WASIp3+ filesystem interface as an explicit diff

WASIp3 testing

Not component-model tests; those are elsewhere

Not testing composition: testing WASI interfaces

A bit of toolchain churn, things settling down now

Done: Clocks, random, filesystem, HTTP*

To do: sockets, sockets+http; use wasi-testsuite python to drive HTTP server; merge pending tests

WASIp3 testing

“Does my WASI implementation correctly implement WASIp3?”

`wasi-testsuite` should have the answer!

`https://github.com/WebAssembly/wasi-testsuite`