# Vitest Vite-native test runner

**START HERE**

`npx vitest`
Watch mode (run mode in CI)

`npx vitest --ui`
Open the browser UI

`npx vitest run`
Run once and exit

`npx vitest related <file> --run`
Tests touching a file

`npx vitest <filter>`
Only files matching a path

`npx vitest -t "<name>"`
Only tests matching a name

`npx vitest run --coverage`
Run with coverage

`npx vitest -u`
Update snapshots

**Filter** path, -t, --changed → `select` → **Run** workers, pool, env → `report` → **Report** --reporter, --coverage → `iterate` → **Watch** a / f / u / p / t keys

## CONCEPTS

Plain vitest watches when run in a terminal and runs once when in CI or without a TTY. Use vitest run for a guaranteed single run.

CLI flags override the config file (vitest.config.ts or the test block in vite.config.ts); keep settings there and use flags for one-off runs.

## RUN TESTS

`npx vitest`
Start watch mode (or one run in CI)

`npx vitest run`
Run once, no watch

`npx vitest --run`
Same as run, for any command

`npx vitest <filter> <filter>`
Run files whose path contains a filter
*Plain substring, not a glob or regexp. Several filters are OR-ed.*

`npx vitest <file>:<line>`
Run the test at a line number
*Single line only; no ranges.*

`npx vitest -t "<pattern>"`
Run tests whose full name matches a regexp

`npx vitest run <file> -t "<pattern>"`
Combine file and name filters

## RUN TESTS (cont.)

`npx vitest related <file> --run`
Run tests that import the given files
*Static imports only. Handy in lint-staged hooks.*

`npx vitest run --changed`
Run tests affected by uncommitted changes

`npx vitest run --changed HEAD~1`
Tests affected since a commit or branch

`npx vitest run --exclude <glob>`
Skip additional files

`npx vitest run --passWithNoTests`
Do not fail when nothing matches

## WATCH MODE KEYS

`a / Enter`
Rerun all tests

`f`
Rerun only failed tests

`r`
Rerun the current pattern

`u`
Update failing snapshots

`p`
Filter by file name

`t`
Filter by test name regexp

## WATCH MODE KEYS (cont.)

`h`
Show the key help

`q`
Quit

## OUTPUT

`npx vitest run --reporter=verbose`
One line per test

`npx vitest run --reporter=dot`
Compact dots

`npx vitest run --reporter=json --outputFile=<file>`
Write JSON results to a file

`npx vitest run --reporter=junit --outputFile=<file>`
JUnit XML for CI

`npx vitest run --reporter=default --reporter=github-actions`
Combine reporters
*The flag is repeatable. Others: tap, tree, blob, minimal, hanging-process.*

`npx vitest run --silent=passed-only`
Show console output of failures only

`npx vitest run --expandSnapshotDiff`
Full diff when a snapshot fails

## OUTPUT (cont.)

`npx vitest list`
Print the collected tests without running

`npx vitest list --filesOnly --json`
List test files as JSON

`npx vitest --ui`
Open the browser dashboard
*Needs the @vitest/ui package. Add --open=false to skip the browser.*

## COVERAGE

`npx vitest run --coverage`
Run with coverage report
*Needs @vitest/coverage-v8 or @vitest/coverage-istanbul installed.*

`npx vitest run --coverage.provider=istanbul`
Choose v8 or istanbul

`npx vitest run --coverage --coverage.reporter=html`
Pick report formats

`npx vitest run --coverage --coverage.include="src/**"`
Count uncovered files too
*By default only files loaded by tests appear.*

`npx vitest run --coverage --coverage.exclude=<glob>`
Leave files out of the report

`npx vitest run --coverage --coverage.thresholds.lines=80`
Fail below 80% lines

## SNAPSHOTS AND FAILURES

`npx vitest run -u`
Rewrite all failing snapshots

`npx vitest run --bail 1`
Stop after the first failure

`npx vitest run --retry 2`
Retry failed tests twice

`npx vitest run --testTimeout 20000`
Per-test timeout in ms (default 5000)

`npx vitest run --sequence.shuffle.tests --sequence.seed <n>`
Shuffle tests with a fixed seed

## SPEED AND CI

`npx vitest run --shard 1/3`
Run one third of the suite

`npx vitest run --reporter=blob --shard 1/3`
Save a blob report per shard

`npx vitest --mergeReports`
Merge blob reports from shards
*Reads .vitest/blob by default; pass a directory otherwise.*

`npx vitest run --pool=threads`
Use worker threads instead of forks

`npx vitest run --maxWorkers 2`
Limit parallel workers

## SPEED AND CI (cont.)

`npx vitest run --no-file-parallelism`
Run files one after another

`npx vitest run --project <name>`
Run one workspace project
*Repeatable; supports wildcards and ! negation.*

## ENVIRONMENT AND DEBUGGING

`npx vitest run --environment=jsdom`
Test in a DOM environment
*jsdom or happy-dom must be installed. Default is node.*

`npx vitest --inspectBrk --no-file-parallelism`
Debug with the Node inspector
*Attach from Chrome or VS Code at 127.0.0.1:9229.*

## RECIPES

**CI run with reports and coverage**

```
npx vitest run --coverage \
  --reporter=default --
reporter=junit \
  --outputFile.junit=reports/
junit.xml
```

**Pre-commit: tests for staged files**

```
git diff --name-only --cached \
  | xargs npx vitest related --
run
```