# Playwright Browser test runner

**START HERE**

**`npm init playwright@latest`**
Set up a project

**`npx playwright install`**
Download browsers

**`npx playwright test`**
Run all tests

**`npx playwright test --ui`**
Pick and watch tests

**`npx playwright test --debug`**
Step through a test

**`npx playwright codegen <url>`**
Record a test

**`npx playwright show-report`**
Open the HTML report

**`npx playwright test --last-failed`**
Re-run failures

**Config** projects, browsers → `test` → **Workers** parallel files → `reporter` → **Report** HTML, traces → `show-trace` → **Trace viewer** timeline, DOM

## CONCEPTS

Test filters are regexes on the file path, so "login" runs every file with login in its path.

Each project in the config is one browser or device setup. Select it with --project.

By default the config records a trace only on the first retry. Use --trace on to force it locally.

## SETUP

**`npm init playwright@latest`**
Scaffold config, example tests and CI file

**`npx playwright install`**
Install the default browsers

**`npx playwright install --with-deps`**
Install browsers and system libraries
*Needs sudo on Linux; the usual step in CI images.*

**`npx playwright install <browser>`**
Install one browser (chromium, firefox, webkit)

**`npx playwright clear-cache`**
Clear build and test caches

**`npm install -D @playwright/test@latest`**
Update Playwright

## RUNNING TESTS

**`npx playwright test`**
Run every test headless

**`npx playwright test <file>`**
Run files matching a path regex

**`npx playwright test <file>:<line>`**
Run the test at a line

**`npx playwright test -g "<title>"`**
Run tests whose title matches

**`npx playwright test -G "<title>"`**
Skip tests whose title matches

## RUNNING TESTS (cont.)

**`npx playwright test --project=<name>`**
Run one project (browser)

**`npx playwright test --headed`**
Show the browser window

**`npx playwright test --last-failed`**
Re-run only the failures

**`npx playwright test --only-changed`**
Run files changed since HEAD

**`npx playwright test --list`**
List tests without running

**`npx playwright test -x`**
Stop after the first failure

**`npx playwright test -c <file>`**
Use another config file

## DEBUGGING

**`npx playwright test --ui`**
Interactive UI mode with time travel

**`npx playwright test --debug`**
Run with the Inspector, headed, 1 worker

**`npx playwright test --trace on`**
Record a trace for every test

**`npx playwright test --trace retain-on-failure`**
Keep traces only for failures

**`npx playwright show-trace <trace.zip>`**
Open a trace in the viewer

**`npx playwright test --timeout <ms>`**
Per-test timeout (default 30000)

**`npx playwright test --repeat-each <N>`**
Run each test N times (find flakes)

## PARALLELISM AND CI

**`npx playwright test --workers <N>`**
Number of workers or a percent of cores

## PARALLELISM AND CI (cont.)

**`npx playwright test --workers=1`**
Run serially in one worker

**`npx playwright test --fully-parallel`**
Run tests of one file in parallel too

**`npx playwright test --retries <N>`**
Retry failed tests N times

**`npx playwright test --shard <n>/<total>`**
Run one slice, e.g. 2/4

**`npx playwright test --forbid-only`**
Fail if test.only is left in

**`npx playwright test --fail-on-flaky-tests`**
Fail when a test only passed on retry

**`npx playwright merge-reports <dir>`**
Merge sharded blob reports

## REPORTS AND OUTPUT

**`npx playwright show-report`**
Open the last HTML report

**`npx playwright show-report <dir>`**
Open a report folder or zip

**`npx playwright test --reporter <name>`**
Choose reporters: list, line, dot, html, json, junit

**`npx playwright test --reporter=html,list`**
Combine reporters, comma separated

**`npx playwright test --output <dir>`**
Folder for artifacts (test-results)

## SNAPSHOTS

**`npx playwright test -u`**
Update snapshots that changed

**`npx playwright test -u all`**
Rewrite every snapshot

## SNAPSHOTS (cont.)

**`npx playwright test --update-snapshots missing`**
Only write absent snapshots

## CODEGEN

**`npx playwright codegen <url>`**
Record actions into test code

**`npx playwright codegen -o <file>`**
Save the generated script

**`npx playwright codegen --target <lang>`**
Pick the language, e.g. python

**`npx playwright codegen -b <browser>`**
Record in firefox or webkit

**`npx playwright codegen --device "<name>"`**
Emulate a device

**`npx playwright codegen --save-storage <file>`**
Save logged-in state
*The file holds cookies and tokens; keep it out of git.*

**`npx playwright codegen --load-storage <file>`**
Start from saved state

## RECIPES

**Debug one failing test**

```
# run the test at line 42 with the Inspector
npx playwright test tests/login.spec.ts:42 --debug
```

**Reuse a login**

```
# log in once and save the state
npx playwright codegen --save-storage=auth.json <url>
# later sessions start logged in
npx playwright codegen --load-storage=auth.json <url>
```

**Shard in CI and merge**

```
# on each of 3 machines
npx playwright test --shard=1/3 --reporter=blob
# then on one machine, with all blobs in one folder
npx playwright merge-reports --reporter html ./all-blob-reports
```