Jest JavaScript test runner

Find tests testMatch → transform → **Run in workers** per file → expect → **Report** results, coverage

CONCEPTS

Through npm, put -- before flags: npm test -- -t "login". Flags override the config file.

RUN TESTS

npx jest
Run every test

npx jest *<pattern>*
Run test files matching a regex

npx jest -t "*<name>***"**
Run tests whose name matches

npx jest --watch
Rerun tests related to changes

npx jest --watchAll
Rerun all tests on any change

npx jest --onlyChanged
Only tests for uncommitted changes

npx jest --bail
Stop after the first failing suite

OUTPUT AND SPEED

npx jest --coverage
Collect and print coverage

OUTPUT AND SPEED (cont.)

npx jest --verbose
List every test with its result

npx jest --ci
Never write new snapshots

npx jest --runInBand
Run serially in one process

npx jest --maxWorkers=*<n>*
Limit parallel workers (n or 50%)

SNAPSHOTS

npx jest -u
Re-record failing snapshots

npx jest -u -t "*<name>***"**
Update snapshots of matching tests only

expect(*<value>***).toMatchSnapshot()**
Compare against a stored snapshot

WRITING TESTS

test("*<name>***", () => { ... })**
Define a test (alias: it)

describe("*<name>***", () => { ... })**
Group tests

WRITING TESTS (cont.)

test.only("*<name>***", fn)**
Run only this test in the file

test.skip("*<name>***", fn)**
Skip this test

test.each([[a, b], ...])("*<name>***", fn)**
Run one test per table row

beforeEach(fn) / afterEach(fn)
Run before or after every test

expect(*<value>***).toBe(***<x>***)**
Strict equality (Object.is)

expect(*<value>***).toEqual(***<x>***)**
Deep equality

expect(*<value>***).toContain(***<x>***)**
Array item or substring

expect(*<fn>***).toThrow(***<msg>***)**
Function throws

await expect(*<promise>***).resolves.toBe(***<x>***)**
Assert on a resolved promise

await expect(*<promise>***).rejects.toThrow()**
Assert on a rejected promise

WRITING TESTS (cont.)

Async tests must return or await the promise, or Jest finishes before the assertions run.

MOCKS

jest.fn()
Create a mock function

jest.spyOn(*<obj>***, "***<method>***")**
Spy on a method, keep the original

jest.mock("*<module>***")**
Auto-mock a module (hoisted)

jest.mock("*<module>***", () => ({...}))**
Mock a module with a factory

mockFn.mockReturnValue(*<x>***)**
Always return x

mockFn.mockResolvedValue(*<x>***)**
Return a promise resolving to x

mockFn.mockImplementation(*<fn>***)**
Set the implementation

MOCKS (cont.)

toHaveBeenCalledWith(*<args>***)**
Assert on call arguments

expect(mockFn).toHaveBeenCalledTimes(*<n>***)**
Assert on call count

jest.useFakeTimers()
Fake setTimeout and friends

jest.advanceTimersByTime(*<ms>***)**
Move fake time forward

CONFIG

testEnvironment: "jsdom"
Browser-like DOM (needs jest-environment-jsdom)

RECIPES

Debug one failing test

```
npx jest path/to/file.test.js -t "<name>" --runInBand
# attach a debugger:
node --inspect-brk node_modules/.bin/jest --runInBand
```

Pre-commit: test what you changed

```
npx jest --bail --findRelatedTests src/a.js src/b.js
```