

THAM KHẢO NHANH JEST

Tests, matchers, mocking, async, và snapshots

Cài Đặt

```
Cài Đặt
npm install --save-dev jest
# package.json: "scripts": { "test": "jest" }
np test # run all tests
np test --watch # re-run on changes
```

Đặt Tên File

- *.test.js** File test (mẫu mặc định)
- *.spec.js** Mẫu test thay thế
- __tests__/*** Thư mục test (tự động phát hiện)

Chạy Test Cụ Thể

```
np test path/to/file.test.js
np test --testNamePattern="adds"
np test --verbose # detailed output
```

Tests Cơ Bản

```
Cấu Trúc Test
describe("Calculator", () => {
  test("adds 1 + 2 to equal 3", () => {
    expect(add(1, 2)).toBe(3);
  });
});
```

test vs it

```
test("works correctly", () => { /* ... */ });
it("should work correctly", () => { /* ... */ });
// Both are identical; "it" reads like English
```

Bỏ Qua & Tập Trung

- test.skip()** Bỏ qua test này
- test.only()** Chỉ chạy test này
- describe.skip()** Bỏ qua toàn bộ suite
- describe.only()** Chỉ chạy suite này

Matchers

Bảng Nhau

- .toBe(val)** Bảng chặt chẽ (===)
- .toEqual(val)** Bảng sâu (object/mảng)
- .toStrictEqual(val)** Sâu + kiểu + thuộc tính undefined
- .not.toBe(val)** Phủ định bất kỳ matcher nào

Tính Đúng Sai

- .toBeTruthy()** Giá trị truthy
- .toBeFalsy()** Giá trị falsy
- .toBeNull()** Chính xác là `null`
- .toBeUndefined()** Chính xác là `undefined`
- .toBeDefined()** Không phải `undefined`

Số

- .toBeGreaterThan(n)** Lớn hơn n
- .toBeLessThanOrEqual(n)** Nhỏ hơn hoặc bằng
- .toBeCloseTo(0.3, 5)** So sánh float (5 chữ số)

Chuỗi, Mảng, Object

- .toMatch(regex)** Chuỗi khớp regex
- .toContain(item)** Mảng/iterable chứa item
- .toHaveLength(n)** Độ dài mảng/chuỗi
- .toHaveProperty(key, val)** Object có thuộc tính
- .toMatchObject(obj)** Object chứa tập con

Test Bất Đồng Bộ

```
async / await
test("fetches data", async () => {
  const data = await fetchData();
  expect(data).toEqual({ id: 1 });
});
```

Promises

```
test("resolves to data", () => {
  return expect(fetchData())
    .resolves.toEqual({ id: 1 });
});
```

Rejections

```
test("rejects with error", async () => {
  await expect(fetchBad())
    .rejects.toThrow("Not Found");
});
```

Exceptions

```
test("throws on invalid input", () => {
  expect(() => validate(null)).toThrow();
  expect(() => validate(null)).toThrow("invalid");
});
```

Mocking

Mock Functions

```
const fn = jest.fn();
fn("hello");
expect(fn).toHaveBeenCalledWith("hello");
expect(fn).toHaveBeenCalledTimes(1);
```

Giá Trị Trả Về Mock

```
const fn = jest.fn()
  .mockReturnValue(42)
  .mockReturnValueOnce(99);
fn(); // 99 (first call)
fn(); // 42 (subsequent)
```

Mock Modules

```
jest.mock("./api");
const { fetchUser } = require("./api");
fetchUser.mockResolvedValue({ name: "Alice" });
```

Mock Matchers

- .toHaveBeenCalled()** Đã gọi ít nhất một lần
- .toHaveBeenCalledTimes(n)** Đã gọi chính xác n lần
- .toHaveBeenCalledWith(args)** Đã gọi với arguments cụ thể

.toHaveBeenLastCalledWith(args) Lần gọi cuối có arguments này

Spies

```
Theo Dõi Methods
const spy = jest.spyOn(Math, "random")
  .mockReturnValue(0.5);
expect(Math.random()).toBe(0.5);
spy.mockRestore(); // restore original
```

Theo Dõi Object Methods

```
const obj = { greet: () => `Hi ${n}` };
const spy = jest.spyOn(obj, "greet");
obj.greet("Alice");
expect(spy).toHaveBeenCalledWith("Alice");
```

Snapshots

```
Snapshot Testing
test("renders correctly", () => {
  const tree = render(
```

Inline Snapshots

```
test("formats name", () => {
  expect(formatName("alice"))
    .toMatchInlineSnapshot("Alice");
});
```

Cập Nhật Snapshots

```
np test --updateSnapshot # update all
np test --updateSnapshot --testNamePattern="renders"
```

Setup & Teardown

Lifecycle Hooks

```
beforeAll(() => { /* once before all tests */ });
afterAll(() => { /* once after all tests */ });
beforeEach(() => { /* before each test */ });
afterEach(() => { /* after each test */ });
```

Phạm Vi

```
describe("Database", () => {
  beforeEach(() => db.connect());
  afterEach(() => db.disconnect());
  test("reads data", () => { /* ... */ });
});
```

Hooks bên trong describe chỉ áp dụng cho block đó

Cấu Hình

```
jest.config.js
module.exports = {
  testEnvironment: "node",
  coverageThreshold: {
    global: { branches: 80, lines: 80 }
  },
};
```

Tùy Chọn Phổ Biến

- testEnvironment** `node` hoặc `jsdom` (DOM)
- roots** Thư mục tìm kiếm tests
- collectCoverage** Bật báo cáo coverage
- coverageDirectory** Thư mục output cho coverage
- moduleNameMapper** Alias đường dẫn (vd: tiền tố @/)
- transform** Biến đổi file (Babel, TS, v.v.)
- setupFilesAfterFramework** Chạy setup trước mỗi suite

Coverage

```
np test --coverage
np test --collectCoverageFrom="src/**/*.js"
```

Mẫu Phổ Biến

Test Gọi API

```
jest.mock("./api");
test("loads users", async () => {
  api.getUsers.mockResolvedValue({id: 1});
  const users = await loadUsers();
  expect(users).toHaveLength(1);
});
```

Mock Timer

```
jest.useFakeTimers();
test("delays execution", () => {
  const cb = jest.fn();
  setTimeout(cb, 1000);
  jest.advanceTimersByTime(1000);
  expect(cb).toHaveBeenCalled();
});
```

Test Tham Số Hóa

```
test.each([
  [1, 1, 2],
  [2, 3, 5],
])("add(%i, %i) = %i", (a, b, expected) => {
  expect(add(a, b)).toBe(expected);
});
```

Custom Matchers

```
expect.extend({
  toBeWithinRange(received, floor, ceil) {
    const pass = received >= floor
      && received <= ceil;
    return { pass, message: () =>
      `expected ${received} in [${floor},${ceil}]` };
  }
});
```