Edit

Jest cheatsheet

A quick overview to Jest, a test framework for Node.js. This guide targets Jest v20.

Quick start

```
/* Add to package.json */
"scripts": {
  "test": "jest"
}
```

```
# Run your tests
npm test -- --watch
```

See: Getting started

Skipping tests

```
describe.skip(···)
it.skip(···) // alias: xit()
```

See: test.skip

Writing tests

```
describe('My work', () => {
  test('works', () => {
    expect(2).toEqual(2)
  })
})
```

See: describe(), test(), expect()

Setup

```
beforeEach(() => { ... })
afterEach(() => { ... })
```

```
beforeAll(() => { ... })
afterAll(() => { ... })
```

See: afterAll() and more

BDD syntax

```
describe('My work', () => {
  it('works', () => {
    ···
  })
})
```

`it` is an alias for `test`. See: test()

Focusing tests

```
describe.only(···)
it.only(···) // alias: fit()
```

See: test.only

Optional flags

`--coverage`	See a summary of test coverage
`--detectOpenHandles`	See a summary of ports that didn't close
`--runInBand`	Run all tests one after the other

Expect

Basic expectations

```
expect(value)
  .not
  .toBe(value)
  .toEqual(value)
  .toBeTruthy()
```

Note that toEqual is a deep equality check. See: expect()

Objects

```
expect(value)
  .toBeInstanceOf(Class)
  .toMatchObject(object)
  .toHaveProperty(keyPath, value)
```

Strings

```
expect(value)
  .toMatch(regexpOrString)
```

Snapshots

```
expect(value)
  .toMatchSnapshot()
  .toMatchInlineSnapshot()
```

Note that `toMatchInlineSnapshot()` requires Prettier to be set up for the project. See: Inline snapshots

Numbers

```
expect(value)
  .toBeCloseTo(number, numDigits)
  .toBeGreaterThan(number)
  .toBeGreaterThanOrEqual(number)
  .toBeLessThan(number)
  .toBeLessThanOrEqual(number)
```

Others

```
expect.extend(matchers)
expect.any(constructor)
expect.addSnapshotSerializer(serializer)

expect.assertions(1)
```

Errors

```
expect(value)
  .toThrow(error)
  .toThrowErrorMatchingSnapshot()
```

Booleans

```
expect(value)
  .toBeFalsy()
  .toBeNull()
  .toBeTruthy()
  .toBeUndefined()
  .toBeDefined()
```

Objects

```
expect(value)
  .toContain(item)
  .toContainEqual(item)
  .toHaveLength(number)
```

More features

Asynchronous tests

```
test('works with promises', () => {
```

Snapshots

```
it('works', () => {
  const output = something()
```

```
    ...
  })
})
```

```
test('works with async/await', async () => {

  ...
})
```

Return promises, or use async/await. See: Async tutorial

Timers

```
jest.useFakeTimers()
```

```
it('works', () => {
  jest.runOnlyPendingTimers()
  jest.runTimersToTime(1000)
  jest.runAllTimers()
})
```

See: Timer Mocks

```
  expect(output).toMatchSnapshot()
})
```

First run creates a snapshot. Subsequent runs match the saved snapshot. See: Snapshot testing

React test renderer

```
import renderer from 'react-test-renderer'
```

```
it('works', () => {



  expect(tree).toMatchSnapshot()
})
```

React's test renderer can be used for Jest snapshots. See: Snapshot test

Mock functions

Mock functions

```
const fn = jest.fn()
```

```
const fn = jest.fn(n => n * n)
```

See: Mock functions

Instances

Assertions

```
expect(fn)
  .toHaveBeenCalled()
  .toHaveBeenCalledTimes(number)
  .toHaveBeenCalledWith(arg1, arg2, ...)
  .toHaveBeenLastCalledWith(arg1, arg2, ...)
```

```
expect(fn)
  .toHaveBeenCalledWith(expect.anything())
  .toHaveBeenCalledWith(expect.any(constructor))
  .toHaveBeenCalledWith(expect.arrayContaining([ values ]))
  .toHaveBeenCalledWith(expect.objectContaining({ props }))
```

```
const Fn = jest.fn()

a = new Fn()
b = new Fn()
```

```
// → [a, b]
```

See: .mock property

Return values

```
const fn = jest.fn(() => 'hello')
```

or:

```
jest.fn().mockReturnValue('hello')
jest.fn().mockReturnValueOnce('hello')
```

```
  .toHaveBeenCalledWith(expect.stringContaining(string))
  .toHaveBeenCalledWith(expect.stringMatching(regexp))
```

Calls

```
const fn = jest.fn()
fn(123)
fn(456)
```

See: .mock property

Mock implementations

```
const fn = jest.fn()
```

```
fn()    // → 1
fn()    // → 2
```

References

http://facebook.github.io/jest/