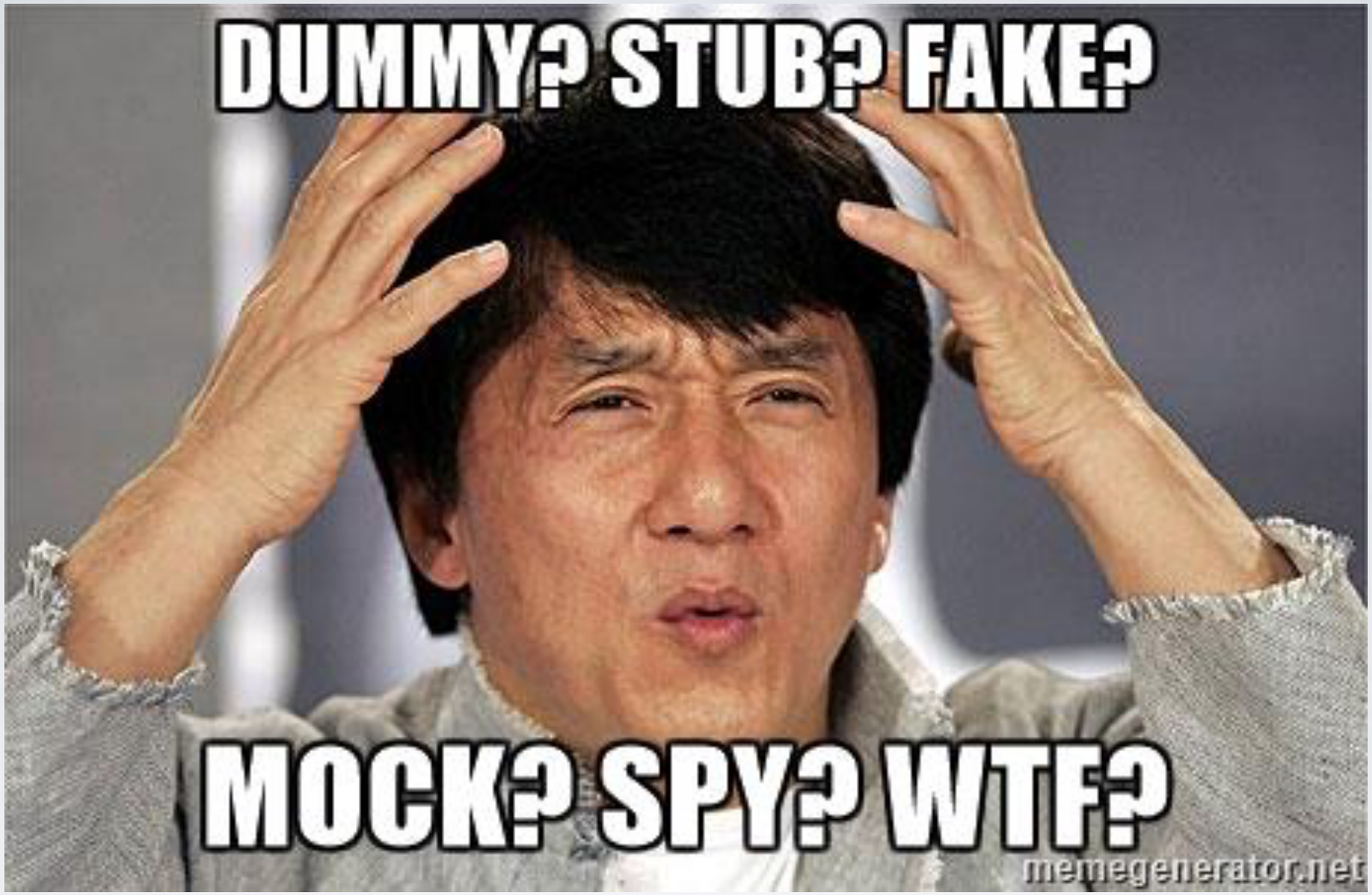

SPYING AND STUBBING

Presented by Eugene Wong

OUTLINE

- Unit testing
- Spies
- Stubs
- Sinon.js

BENEFITS OF UNIT TESTING

- You should know them by now!!!

HOW TO DO UNIT TESTING?

- Ask yourselves these questions:
  - What are you testing?
  - What should it do?
  - What is the actual output?
  - What is the expected output?
  - How can the test be reproduced?

EXAMPLE

```
import test from 'tape';
import compose from '../source/compose';

test('Compose function output type', assert => {
    const actual = typeof compose();
    const expected = 'function';

    assert.equal(actual, expected,
        'compose() should return a function.');

    assert.end();
});
```

TEMPLATE

```
import test from 'tape';

// For each unit test you write,
// answer these questions:
test('What component aspect are you testing?', assert =>
{
    const actual = 'What is the actual output?';
    const expected = 'What is the expected output?';

    assert.equal(actual, expected,
        'What should the feature do?');

    assert.end();
});
```

TEST CASES

- Ideal working case
- Totally not working case
- Multiple other breaking cases

SPIES

- A test spy is an object that records its interaction with other objects throughout the code base.

STUBS

- Test stubs are fake objects with pre-programmed behavior (Simulation of behaviour from other units), Most of times they are simply returning fixed values.

SINON.JS

- DEMO

THANK YOU.