const wrapper = shallow(<MyComponent foo={10} />);

expect(wrapper.prop('foo')).to.equal(10);

**import** React from 'react';

**import** { mount, shallow } from 'enzyme';

**import** {expect} from 'chai';

**import** Avatar from '../lib/avatar';

describe('<Avatar/>', **function** () {

it('should have an image to display the gravatar', **function** () {

**const** wrapper = shallow(<Avatar/>);

expect(wrapper.find('img')).to.have.length(1);

});

it('should have props for email and src', **function** () {

**const** wrapper = shallow(<Avatar/>);

expect(wrapper.props().email).to.be.defined;

expect(wrapper.props().src).to.be.defined;

});

})

Since ReactJS will only send warnings to the console but will not actually throw an error, I test prop values in this way:

var myTestElement = TestUtils.renderIntoDocument(

<MyTestElement height={100} /> );

it("check MyTestElement props", function() {

expect( typeof myTestElement.props.height ).toEqual ( 'number' );

});

it('simulates click events', () **=>** {

**const** onButtonClick **=** sinon.spy();

**const** wrapper **=** shallow(

**<**Foo onButtonClick**=**{onButtonClick} />  );

wrapper.find('button').simulate('click');

expect(onButtonClick.calledOnce).to.be.**true**;

});