METEORでREACT使う意味あんの？
@BESUTOME

ないです

REACT?

https://facebook.github.io/react/index.html

- MVCのViewのみサポート
- Virtual DOM (仮想DOM)
- 一方向バインディング

JQUERY

```
// Submitされたら
$('form').on('submit', function() {
  // 要素作って
  var $li = $('<li>');
  // リストに追加
  $('ul').append($li);
});
```

REACT

```
var Form = React.createClass({
  onSubmit: function() {
    // 親にデータの更新を通知
  },
  render: function() {
    return <form onSubmit={this.onSubmit}>...</form>;
  }
});

var List = React.createClass({
  render: function() {
    // 親からもらったデータを元に構築するだけ
    return <ul>{this.props.list.map(...)</ul>;
  }
});
```

ENZYME?

http://airbnb.io/enzyme/

- airbnbがつくった
- Reactのテストフレームワーク
- いろんな使い方がある

INSTALL

```
$ meteor add react
$ meteor add meteorhacks:npm

{
  "enzyme": "1.2.0"
}
```

SHALLOW RENDERING

- ユニットテスト
- TDDで開発するときはコレ

```
import { shallow } from 'enzyme';

describe('<MyComponent />', () => {

  it('should render three <Foo /> components', () => {
    const wrapper = shallow(<MyComponent />);
    expect(wrapper.find(Foo)).to.have.length(3);
  });

  it('should render an `.icon-star`', () => {
    const wrapper = shallow(<MyComponent />);
    expect(wrapper.find('.icon-star')).to.have.length(1);
  });

  it('should render children when passed in', () => {
    const wrapper = shallow(
      <MyComponent>
        <div className="unique" />
      </MyComponent>
    );
    expect(wrapper.contains(<div className="unique" />)).to.be.true;
  });

  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;
  });

});
```

FULL RENDERING

- DOM呼出に関するテスト
- Static Renderingとの組み合わせ

```
import { mount } from 'enzyme';

describe('<Foo />', () => {

  it('calls componentDidMount', () => {
    spy(Foo.prototype, 'componentDidMount');
    const wrapper = mount(<Foo />);
    expect(Foo.prototype.componentDidMount.calledOnce).to.be.true;
  });

  it('allows us to set props', () => {
    const wrapper = mount(<Foo bar="baz" />);
    expect(wrapper.props().bar).to.equal("baz");
    wrapper.setProps({ bar: "foo" });
    expect(wrapper.props().bar).to.equal("foo");
  });

  it('simulates click events', () => {
    const onButtonClick = spy();
    const wrapper = mount(
      <Foo onButtonClick={onButtonClick} />
    );
    wrapper.find('button').simulate('click');
    expect(onButtonClick.calledOnce).to.be.true;
  });

});
```

STATIC RENDERED MARKUP

- 出力された静的なHTMLに対してのテスト
- Full Renderingとの組み合わせ

```
import { render } from 'enzyme';

describe('<Foo />', () => {

  it('renders three `.foo-bar`s', () => {
    const wrapper = render(<Foo />);
    expect(wrapper.find('.foo-bar').length).to.equal(3);
  });

  it('rendered the title', () => {
    const wrapper = render(<Foo title="unique" />);
    expect(wrapper.text()).to.contain("unique");
  });

});
```

まとめ

- MeteorとReactは設計思想が違う
- Meteor的に一方向バインディングにならない
- Reactはテストしやすい
- Enzyme便利