

React.JS

Ynon Perek

ynon@ynonperek.com

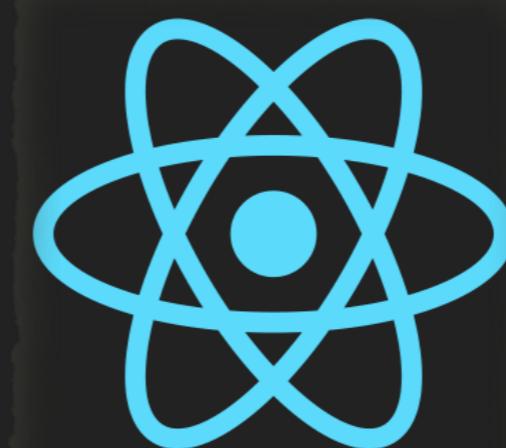


Offer a ***consistent*** and
simple way to write web
applications

Web Application Challenges

- Code architecture
- Split UI to components
- Navigation
- Application State

Client Side Frameworks



BACKBONE.JS



Why React

- Easy to learn
- Carries heavy weight

Who Uses React



React IS NOT

- An MVC framework
(it won't tell you how to manage your data)
- A SPA framework
(it won't handle client-side routing)

Let's See Some Code (our first React component)

```
1 var HelloWorld = React.createClass({  
2   render: function() {  
3     return (  
4       <div>  
5         <h1>Hello React </h1>  
6         <p>Markup + Code = Awesome</p>  
7       </div>  
8     )  
9   }  
10});  
11
```

React How

- Use Babel to convert JSX code to JS
- Use `React.render(...)` to paint a component on screen

React How

- Client side React+TDD starter:
<https://github.com/ynonp/react-tdd-demo>
- Alternatively, fork this pen:
<http://codepen.io/ynonp/pen/VLyzpW?editors=101>

JSX vs. JS

```
var App = React.createClass({  
  render: function() {  
    return <div>  
      Hello React!  
    </div>  
  }  
});  
  
React.render(<App />, document.querySelector('main'));
```

JSX vs. JS

```
'use strict';

var App = React.createClass({
  displayName: 'App',

  render: function render() {
    return React.createElement(
      'div',
      null,
      'Hello React!'
    );
  }
});

React.render(React.createElement(App, null),
  document.querySelector('main'));
```

Customising Components

- Components take properties when created
- That's how outer components affect inner

Customising Components

```
var App = React.createClass({
  render: function() {
    return <div>
      {this.props.message}
    </div>
  }
});

React.render(<App message="Hello World" />,
  document.querySelector('main'));
```

Customising Components

- Can pass any JS object as value for a property
- Remember JSX -> JS

Customising Components

```
var App = React.createClass({
  render: function() {
    return <div>
      I got {this.props.data.length} numbers
    </div>
  }
});

var numbers = [10, 20, 30, 40];

React.render(<App data={numbers} />,
  document.querySelector('main'));
```

Using Properties

- Changing properties after rendering won't repaint your components
- So best to keep your properties immutable

Dynamic Components

- A dynamic component has a “state”
- That state changes during the component’s life cycle
- Can have initial state

Demo: Let's Write a Click Counter

- Internal State: How many clicks performed so far
- External Input: None

Demo: Clicks Counter

```
1 var Counter = React.createClass({
2   getInitialState: function() {
3     return { clicks: 0 };
4   },
5   clicked: function() {
6     this.setState({clicks: this.state.clicks + 1});
7   },
8   render: function() {
9     return (
10       <div>
11         <button onClick={this.clicked}>Click Me</button>
12         <p>You clicked {this.state.clicks} times</p>
13       </div>
14     )
15   }
16 });
17
```

The Good

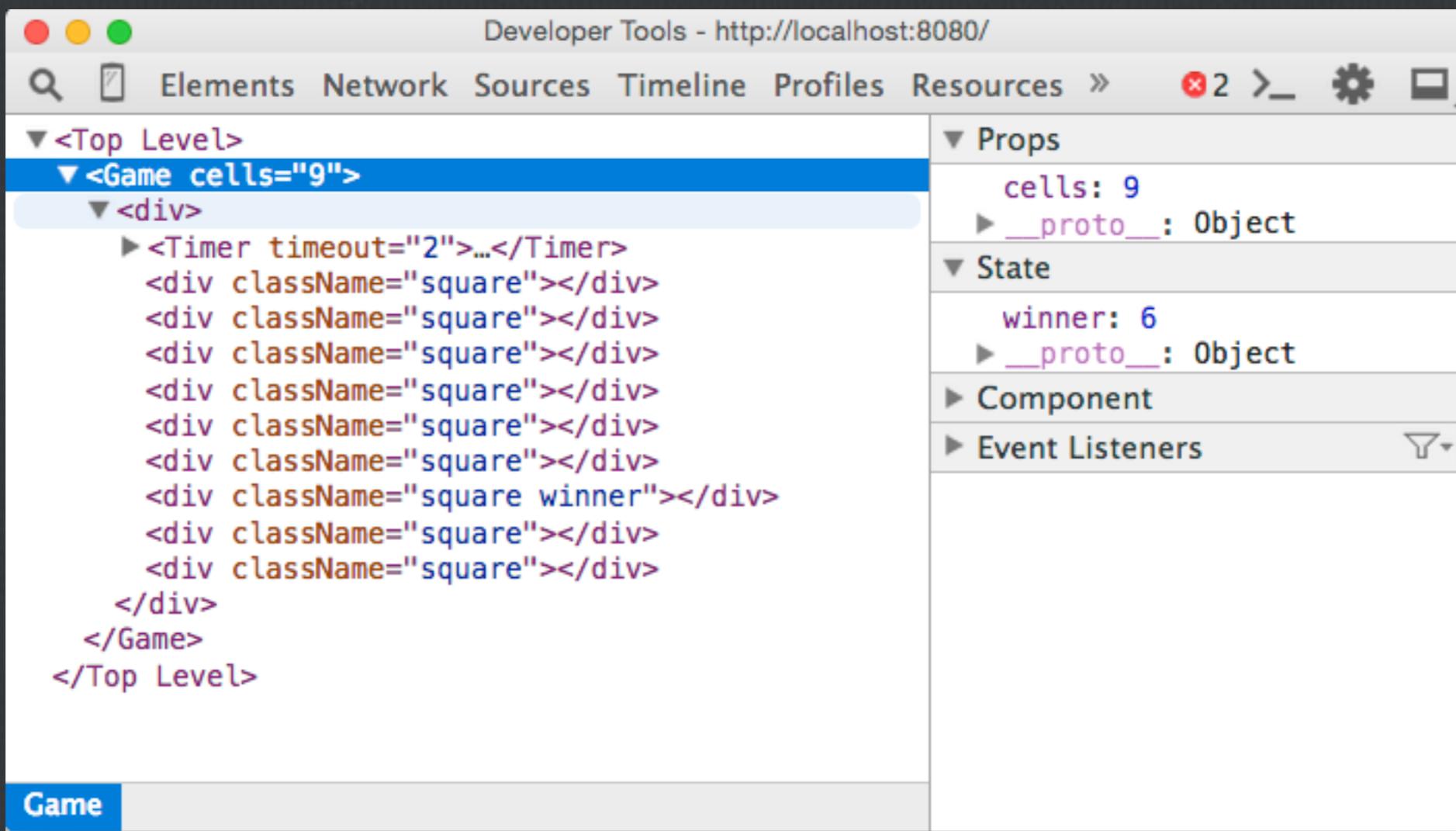
- No need to manipulate DOM nodes
- Combine components to build larger apps

React Debugging

- Webpack creates source maps:
 - Can debug inside chrome devtools
 - Debug with “original” source
 - Demo

React Debugging

□ React Chrome App



React: Behind The Scene

Virtual DOM

form

label

input

label

input

React: Behind The Scene

Virtual DOM

form

label

input

label

input

Virtual DOM

form

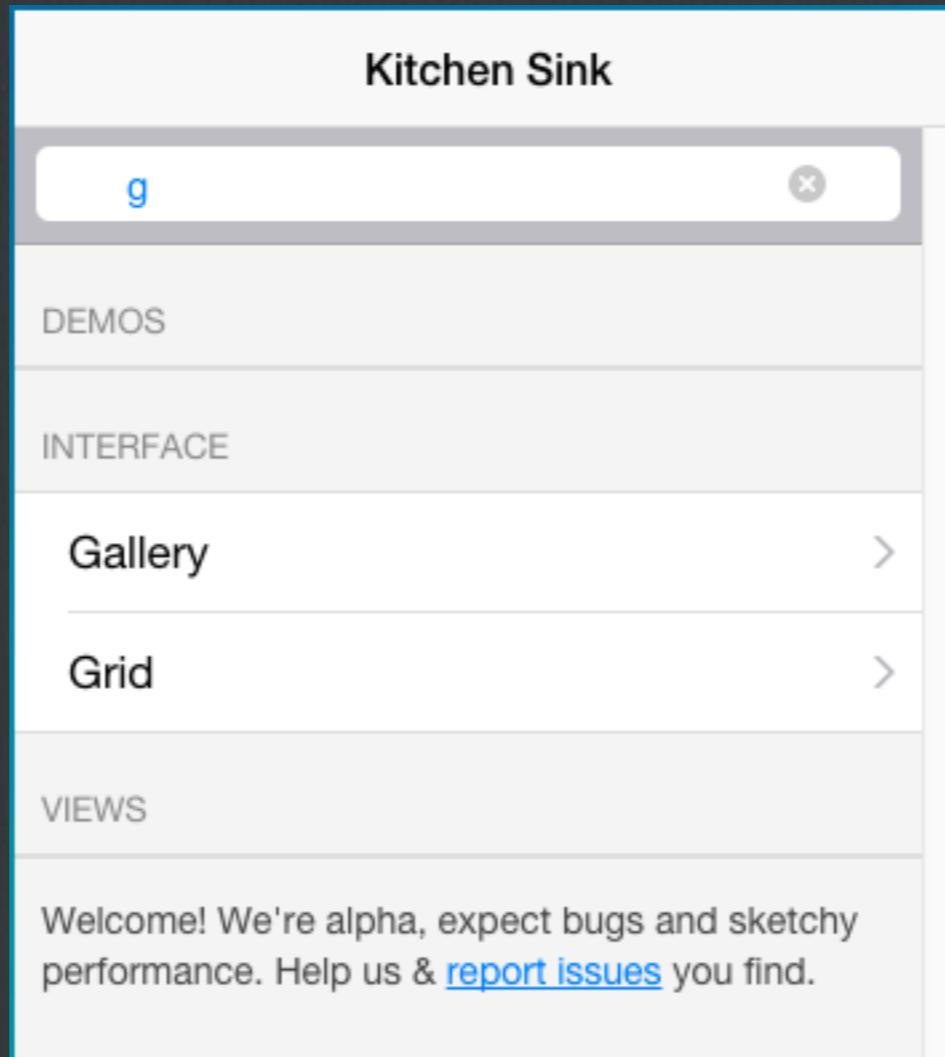
label

input

React: Behind The Scene

- Calculate how to get from virtual DOM tree A to virtual DOM tree B
- In our example:
 - Delete last input and label

Component Based UI



Q & A



Handling External Inputs

- Define properties your component should use
- Example: Let's build a filterable list
 - State: current filter text
 - Properties: array of items to display and filter
- Solution: <http://codepen.io/ynonp/pen/pJbRjV>

React + TDD



Agenda

- TDD short intro
- The tools
- React It: translating old JS to React using TDD
- Conclusions & Takeaways

Hello TDD

- TDD helps you write easy to test code**
- Write tests first**

TDD + React

- React helps us avoid code pitfalls**
- TDD makes sure we get it right**

TDD + React

Markup

Code

Spec

TDD Workflow

The screenshot shows the WebStorm IDE interface with the following details:

- Title Bar:** root-test.js - react-karma - [~/work/courses/talks/react-tdd/react-karma]
- Toolbars:** Standard Java-like toolbar with icons for file operations.
- Project Structure:** Shows the project tree under "react-karma":
 - js
 - node_modules (library home)
 - spec
 - components
 - game-mock-test.js
 - game-test.js
 - root-test.js
 - app-test.js
- Code Editor:** Displays the content of "root-test.js".

```
1 var React = require('react');
2 var TestUtils = require('react/lib/ReactTestUtils');
3 var expect = require('chai').expect;
4 var Root = require('components/root');

5 describe('root', function () {
6     it('renders without problems', function () {
7         var root = TestUtils.renderIntoDocument();
8         expect(root).to.be.a('div');
9     });
10    it('renders children', function () {
11        var root = TestUtils.renderIntoDocument(

hello

);
12        expect(root.innerHTML).to.contain('hello');
13    });
14 });
15 
```
- Run Tab:** Set to karma.conf.js
- Test Results:** Shows test results for "karma.conf.js" in "Chrome 43.0.2357 (Mac OS X 10.10.3)".
 - Test Results: Done: 9 of 9 (0.712 s)
 - Test Cases:
 - app
 - game
 - root
- Output Panel:** Shows the command run and the exit code.

```
/usr/local/bin/node /Applications/WebStorm.app/
Process finished with exit code 0
```
- Bottom Navigation:** Run, TODO, Terminal, Version Control, Changes, Event Log.
- Status Bar:** Tests passed (moments ago), 4:1 LF, UTF-8, Git: master.

TDD Workflow

- Run tests from within IDE, while coding
- Fix code and build more tests as you go along
- Eventually check look & feel in the browser

TDD Tools

- Webstorm**
- Webpack**
- Karma**
- Mocha / Sinon / Chai**
- React.addons.TestTools**

Demo: Vanilla → TDD React

- Given the square game here:
<http://codepen.io/ynonp/pen/YXrPNj?editors=101>
- Let's write a React version using TDD

Start with a test

```
describe('Game cells', function() {
  it('renders .square child nodes', function() {
    var game = TestUtils.renderIntoDocument(<Game cells={9} />);

    var squares = TestUtils.scryRenderedDOMComponentsWithClass(game, "square");
    expect(squares.length).to.eq(9);
  });
});
```

Write Some Code

```
render: function () {
  return (
    <div>
      { _.map(_.range(this.props.cells), this.renderItem)}
    </div>
  )
}
```

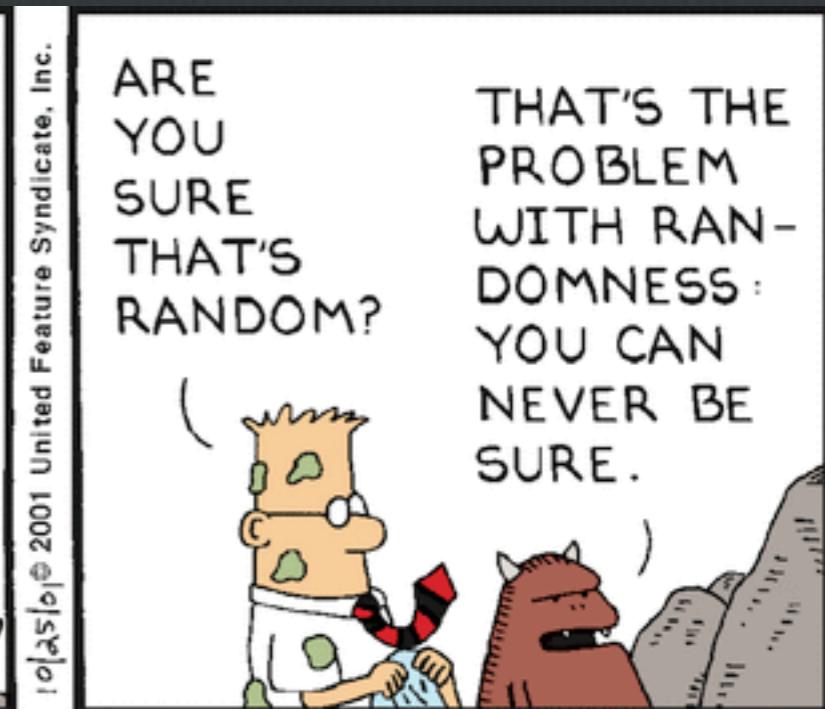
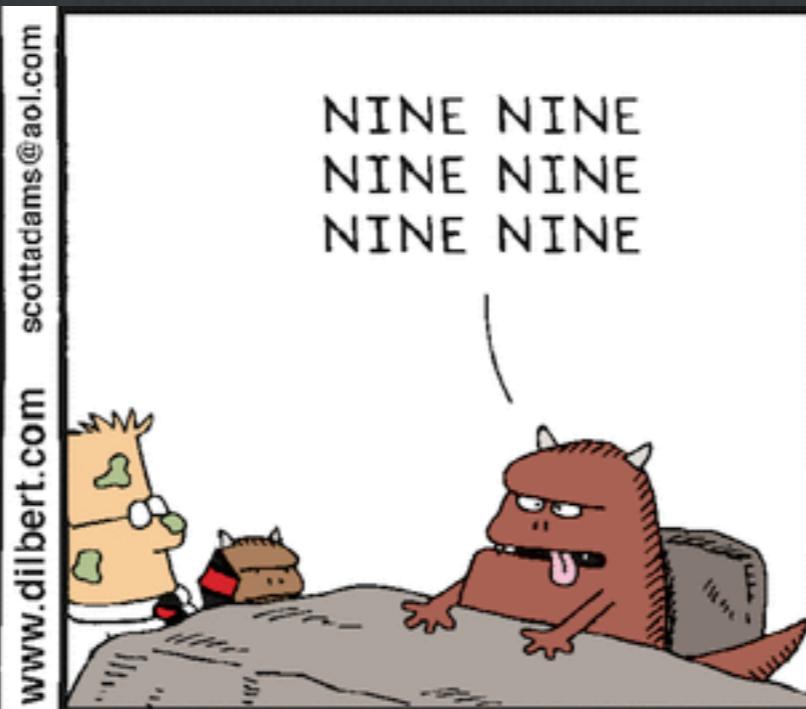
React.addons.TestTools

- **renderIntoDocument**
- **scry / find functions to find React components and DOM nodes**
- **Simulate functions to simulate DOM events**

Simulate and stubs

- Let's randomise the winner
- Let's change the winning square after each click

Problem With Random



Stubbing with sinon

```
it('is selected randomly using _.sample', function() {
  sinon.stub(_, "sample").returns(42);

  var game = TestUtils.renderIntoDocument(<Game cells={3} />);
  expect(game.state.winner).to.eq(42);

  _.sample.restore();
});
```

Other useful stubs

- **Stubbing clocks with sinon.fakeTimers**
- **Stubbing ajax with sinon.fakeServer**

Other useful stubs

- Sub components (rewire)
- Demo: Game Timer

Q & A



Why TDD

- Coding was fun
- The tests are useful going forward
- Better code

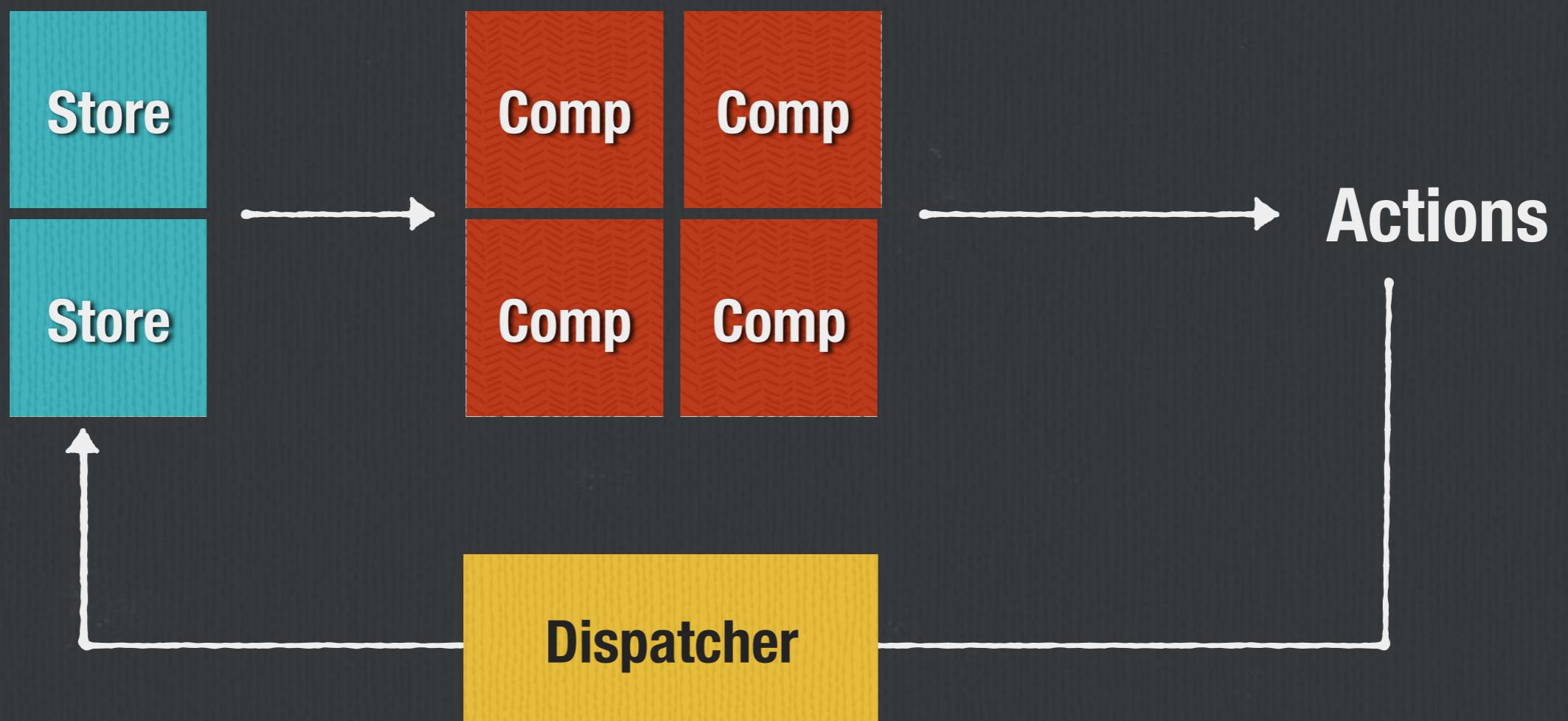
React + Flux Architecture



React Architecture

- No architectural constraints
 - Some write their own
 - Some use MVC (backbone, angular, ember)
 - Some use flux

Flux Architecture



Flux

- **Unidirectional application architecture**
- **Has many implementations: Flux, Reflux, Lux, fluxthis and more**
- **<https://github.com/voronianski/flux-comparison>**

Flux Components

- Stores: Save data
- Dispatcher: handles actions
- Actions: manage flow
- React Components

Flux Demo

- Random message generator:
 - Users store
 - Messages store
- New messages => create new users
- Allow pause/resume using a ListeningStore

Q & A



Thanks For Listening

- Ynon Perek**
- ynon@ynonperek.com**
- www.tocode.co.il**