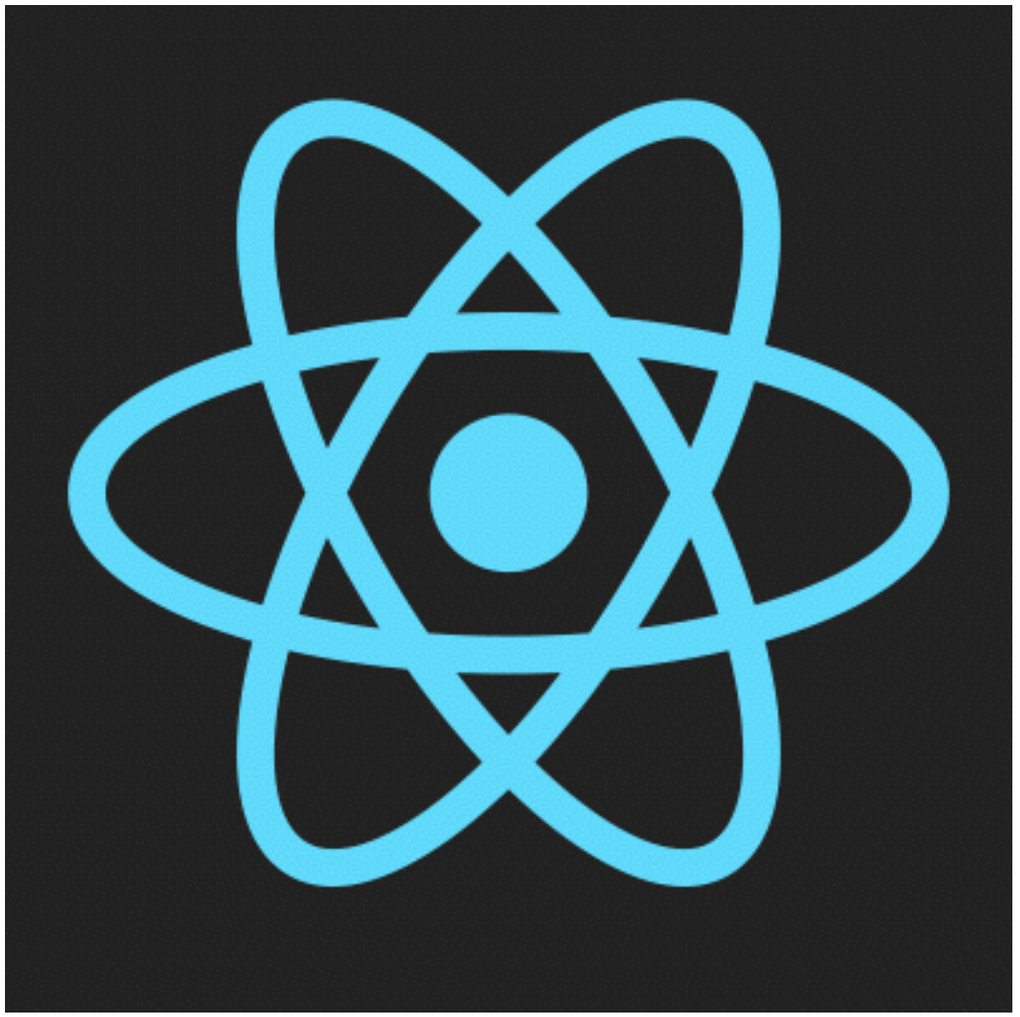
discover REACT



Massimo Iacolare
@iacoware

doubleloop
BETTER SOFTWARE, BETTER BUSINESS

Agenda

1. Introduce REACT

2. Build together a small app to see REACT in action

Hello

REACT

REACT

a library to build

USER INTERFACES

BY FACEBOOK

MAY

2013

MAY

2013 open source

MAY
2013 open source

2011 production

FACEBOOK

FACEBOOK

INSTAGRAM

KHAN ACADEMY

NEW YORK TIMES

AIRBNB

AND MANY OTHERS...

BATTLE TESTED

SKEPTICAL REACTION

Syndrome

Brace YOURSELF

Brace YOURSELF

```
var FirstComponent = React.createClass({
    render: function() {
        return (
            <div>
                <h1>React is awesome</h1>
                <h4>everything will never look the same</h4>
                <button onClick={this.props.onClick}>
                    Pump up the volume
                </button>
            </div>
        );
    }
});
```

Brace YOURSELF

```
var FirstComponent = React.createClass({
    render: function() {
        return (
            <div>
                <h1>React is awesome</h1>
                <h4>everything will never look the same</h4>
                <button onClick={this.props.onClick}>
                    Pump up the volume
                </button>
            </div>
        );
    }
});
```

Wat?

HTML + JS

HTML + JS

==

MIXING CONCERNS

HTML + JS

==

MIXING CONCERNS

Really?

INEVITABLY JS COUPLED TO HTML

```
var model = {
    name: 'A beautiful product',
    image: '...',
    promotionClass: 'promotion-gold'
};
```

INEVITABLY
JS COUPLED TO HTML

```
var model = {
    name: 'A beautiful product',
    image: '...',
    promotionClass: 'promotion-gold'
};
```

HIGH COESION

maybe?

REACT
COMPONENT

HIGH COHESIVE
module

separation of
CONCERNS

separation of

CONCERNS

separation of

COMPONENTS

assumptions will
be challenged

assumptions will
be challenged

keep an

OPEN MIND

Why should I learn

YAWOBS?

Why should I learn

YAWODS?

YET ANOTHER WAY OF DOING STUFF

quest to

BETTER UX

FRONTEND

complexity

today's challenges

PERFORMANCE

ALWAYS

ACCIDENTAL

COMPLEXITY

KEEP
LOW

ACCIDENTAL
COMPLEXITY

DECLARATIVE *over* IMPERATIVE

always seek PREDICTABILITY

EXPLICIT *over* IMPLICIT

REACT

pillars

everything is a

COMPONENT

TODO APP

```
var App = React.createClass({
    render: function() {
        return (
            <div>
                <h1>Awesome Todo</h1>
                <TodoList items={/*...*/}/>
                <NewTodo onAddNew={/*...*/}/>
            </div>
        );
    }
});

var TodoList = React.createClass({
    render: function() {
        return (
            <div>
                {this.props.items.map(item => {
                    return <TodoItem item={item}/>
                })}
            </div>
        );
    }
});

var NewTodo = React.createClass({ /*...*/ });
var TodoItem = React.createClass({ /*...*/ });
```

compose

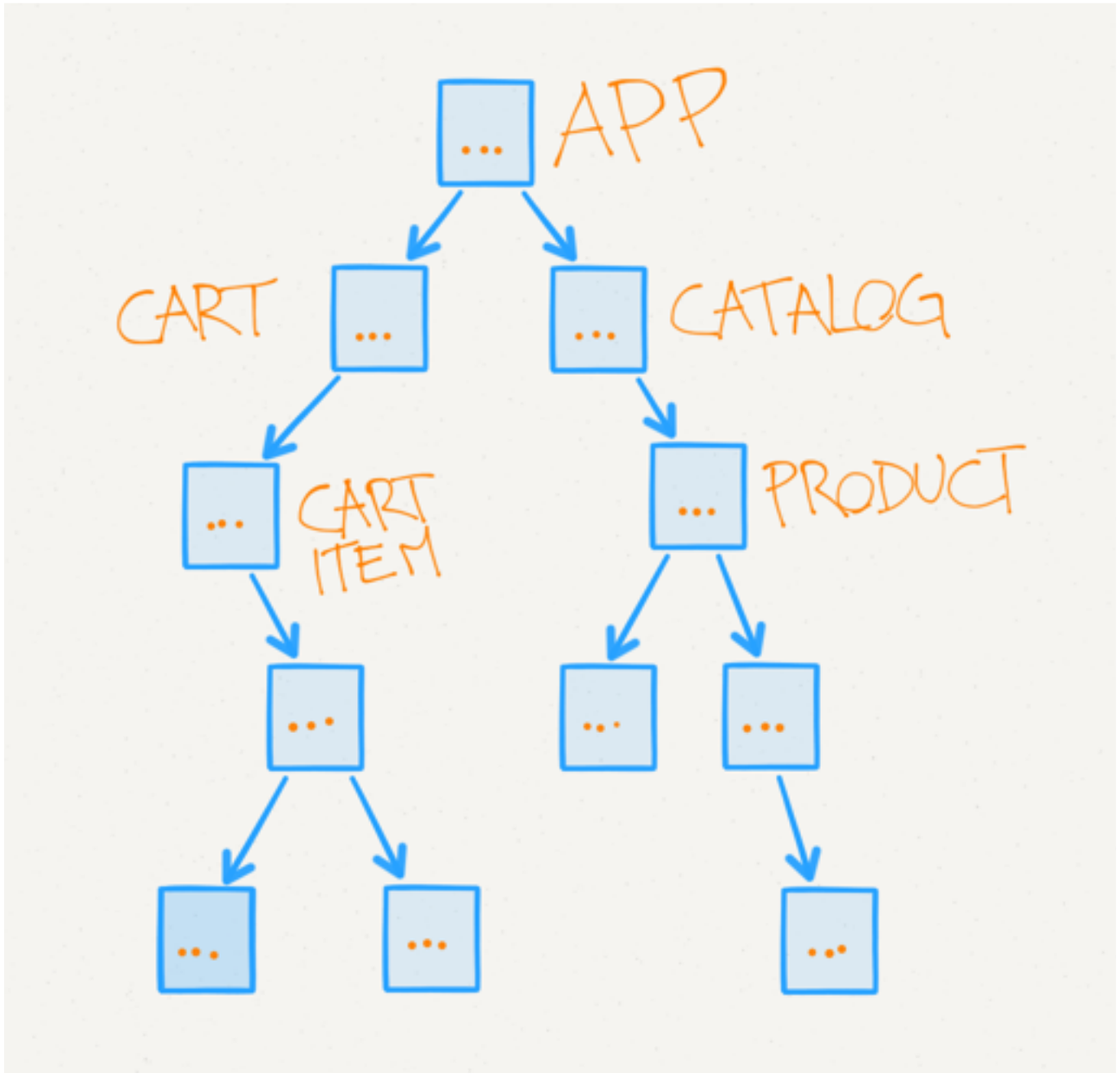
```
<Tab>
    <p>
        This is obviously custom
        and a great way to build
        reusable components!!!
    </p>
</Tab>
```

COMPONENTS

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

COMPONENTS

all the way down



COMPONENT lifecycle

```
componentWillMount()

componentDidMount()

componentWillReceiceProps()

componentWillUpdate()

componentDidUpdate()

componentWillUnmount()
```

trasform

```jsx
<div>
    <span style={...}>{text}</span>
    <input onChange={...} value={text}/>
</div>
```

JSX

into

```jsx
React.createElement('div', null, [
    React.createElement('span', {style: ...}, text),
    React.createElement('input', {onChange: ...})
])
```

to

JS

let's talk about JSX

```
render: function() {
    var text = /*...*/
    return (
        <div>
            <span style={this.computeStyle('show')}>
                {text}
            </span>
            <input
                onChange={this.onChange}
                style={this.computeStyle('edit')}
                value={text}/>
        </div>
    );
}
```

EXPRESSIONS - EVENTS - STYLES

[jsx looks like an abomination](#)

embedded JSX

smaller COMPONENTS

SRP & RESUSABILITY ++

How do we pass data to COMPONENTS?

Two kinds of data

PROPS & STATE

THIS.PROPS

```
render: function() {
    return (
        //more complex component

        <Title
            text="Apericoder"
            hint="where programmers go to drink!" />

        //other stuff
    );
}
```

```
var Title = React.createClass({
    render: function() {
        return (
            <div>
                <h1>{this.props.text}</h1>
                <h4>{this.props.hint}</h4>
            </div>
        );
    }
})
```

optional TYPE CHECK

```
var Title = React.createClass({
    propTypes: {
        text: React.PropTypes.string.isRequired,
        hint: React.PropTypes.string
    },

    render: function() {
        return (
            <div>
                <h1>{this.props.text}</h1>
                <h4>{this.props.hint}</h4>
            </div>
        );
    }
})
```

PROPS

IMMUTABLE

can't touch it

PASSED FROM PARENT

makes tracking down bugs easier

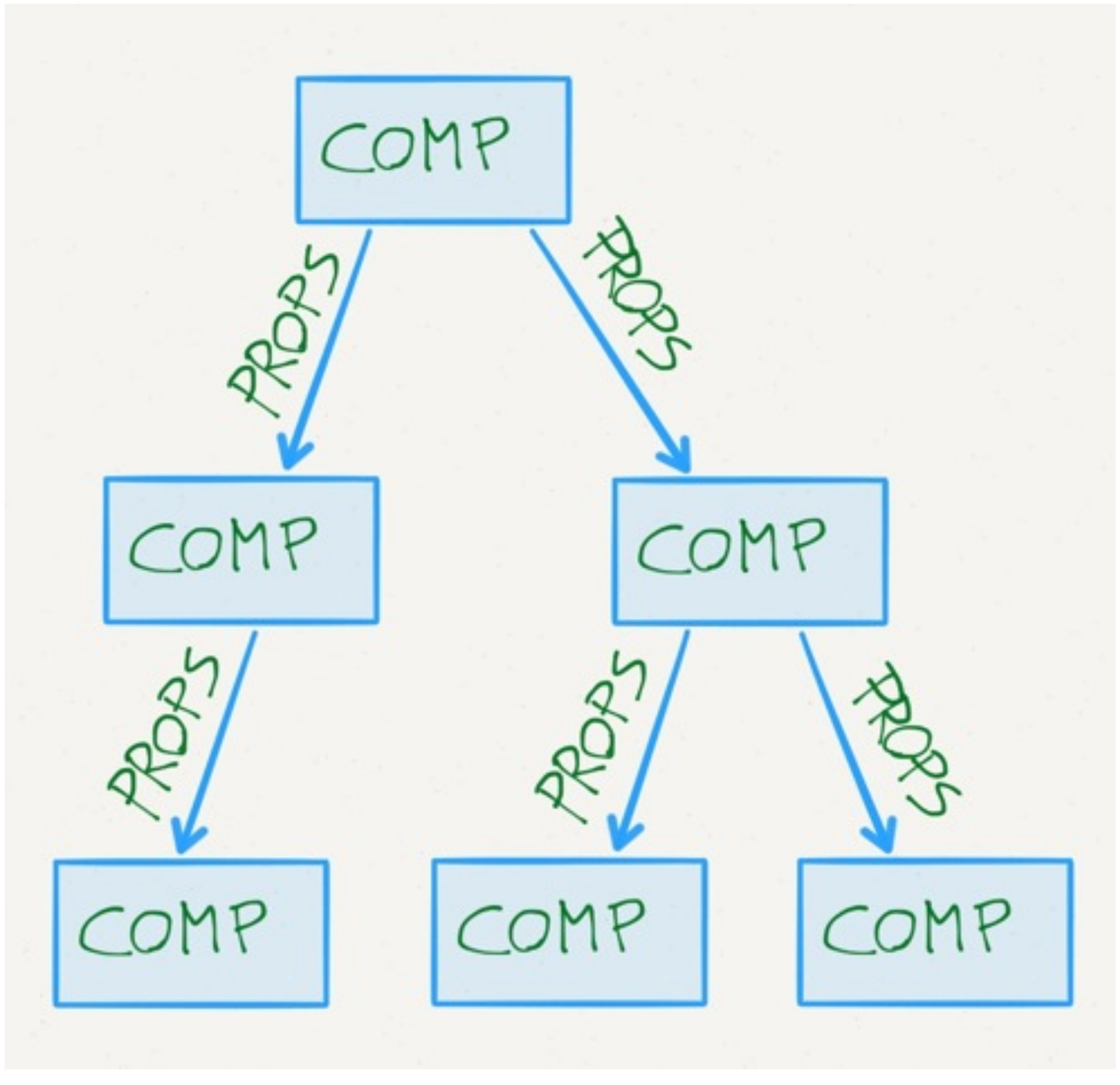
THIS.PROPS TO READ

STATELESS COMPONENT

very easy to reason about

How data flows

How data flows



props passed from parent

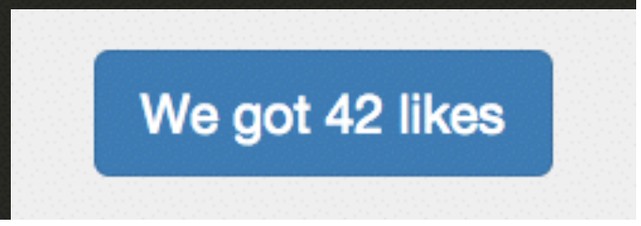
pass callbacks to setup a simple communication mechanism

THIS.STATE

```
var LikeButton = React.createClass({
    getInitialState: function() {
        return { count: 0 };
    },

    increment: function() {
        this.setState({
            count: this.state.count + 1
        });
    },

    render: function() {
        return (
            <button onClick={this.increment}>
                We got {this.state.count} likes
            </button>
        );
    }
})
```



STATE

CREATED INSIDE COMP

MUTABLE

THIS.STATE TO READ
THIS.SETSTATE TO WRITE

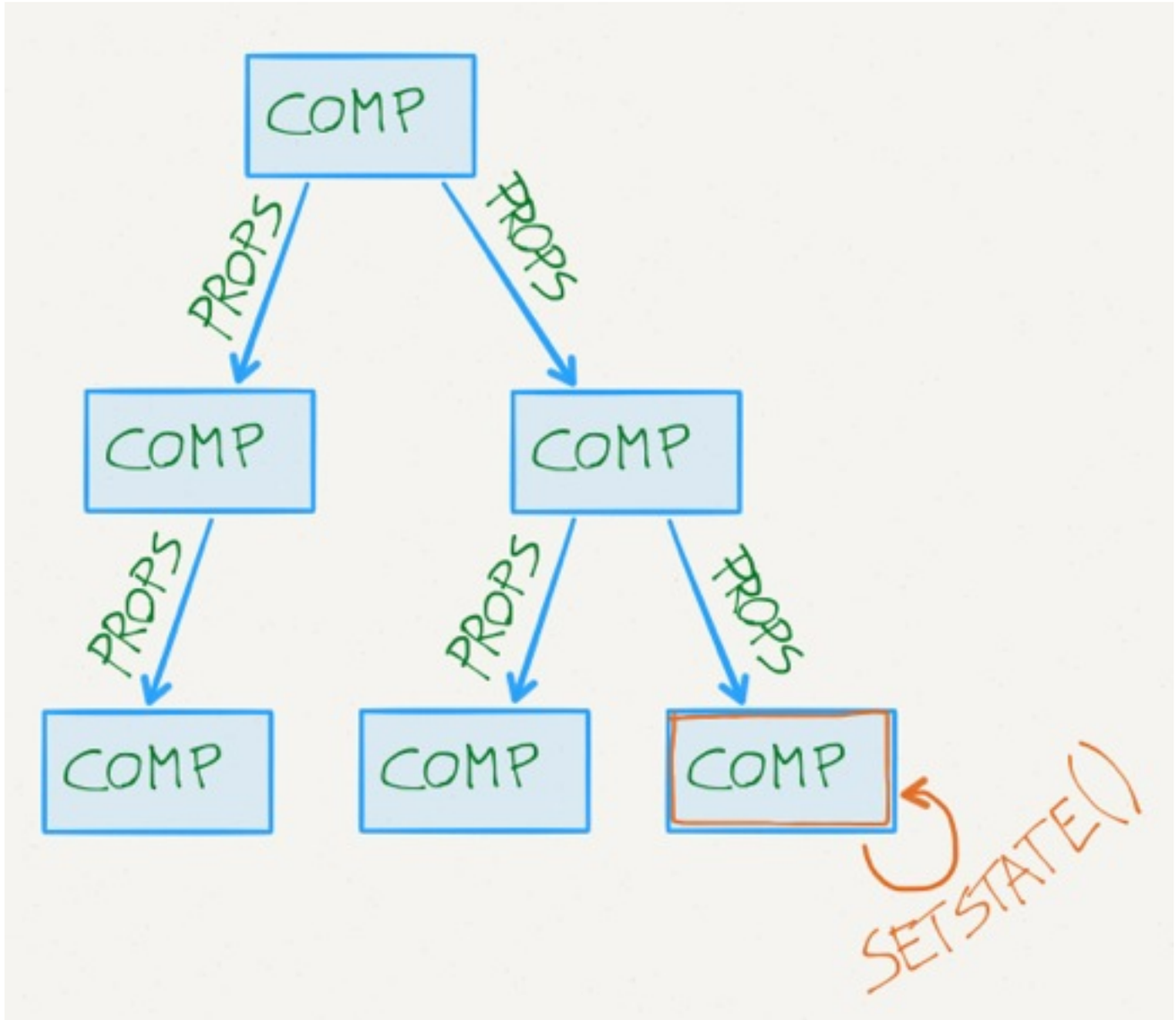
STATEFUL COMPONENT

How data flows

How data flows

state can
become props

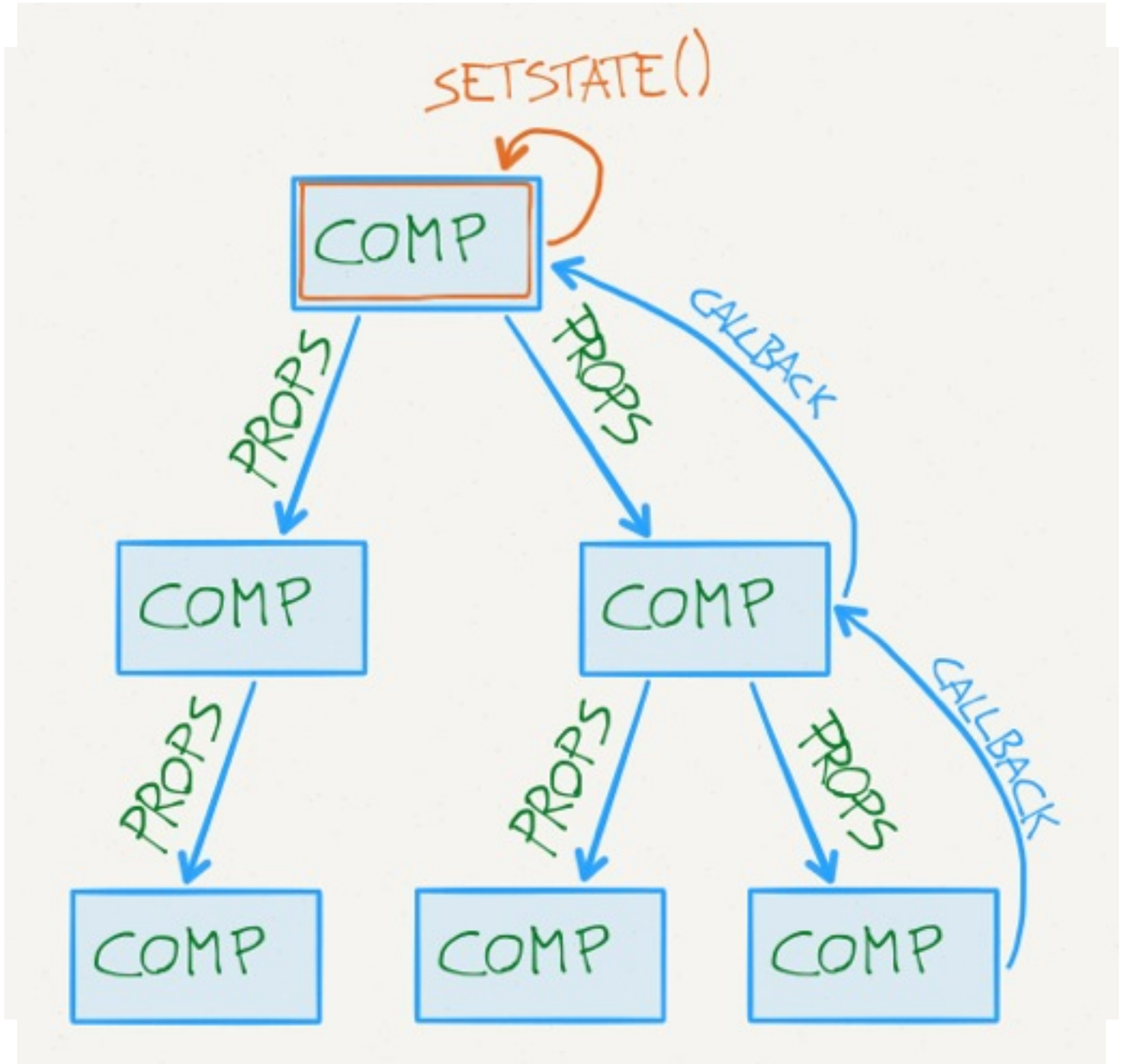
How data flows



state is private
by default

state can
become props

How data flows



state is private
by default

state can
become props

data flows down
callbacks flow up

PROPS

IMMUTABLE

PASSED FROM PARENT

THIS.PROPS TO READ

STATELESS COMP.

STATE

MUTABLE

CREATED INSIDE COMP.

THIS.STATE TO READ
THIS.SETSTATE() TO WRITE

STATEFUL COMP.

why?

because

MUTABLE STATE

is evil

because

MUTABLE STATE

is ~~evil~~

source of complexity

ok cool but...

what happens when DATA CHANGES?

REACT

REACT

doesn't use

TWO-WAY
DATABINDING

TWO-WAY DATABINDING

TWO-WAY DATABINDING

lead to

CASCADING UPDATES

TWO-WAY DATABINDING

TWO-WAY DATABINDING

implict way

MUTATE STATE

remember?

MUTABLE STATE

is evil

remember?

MUTABLE STATE

is ~~evil~~

source of complexity

EVERY TIME setState()

EVERY TIME *setState()*

RE-RENDER *everything*

EVERY TIME *setState()*

RE-RENDER *everything*

SIMPLER *mental model*

EVERY TIME *setState()*

RE-RENDER *everything*

SIMPLER *mental model*

RE-RENDER...

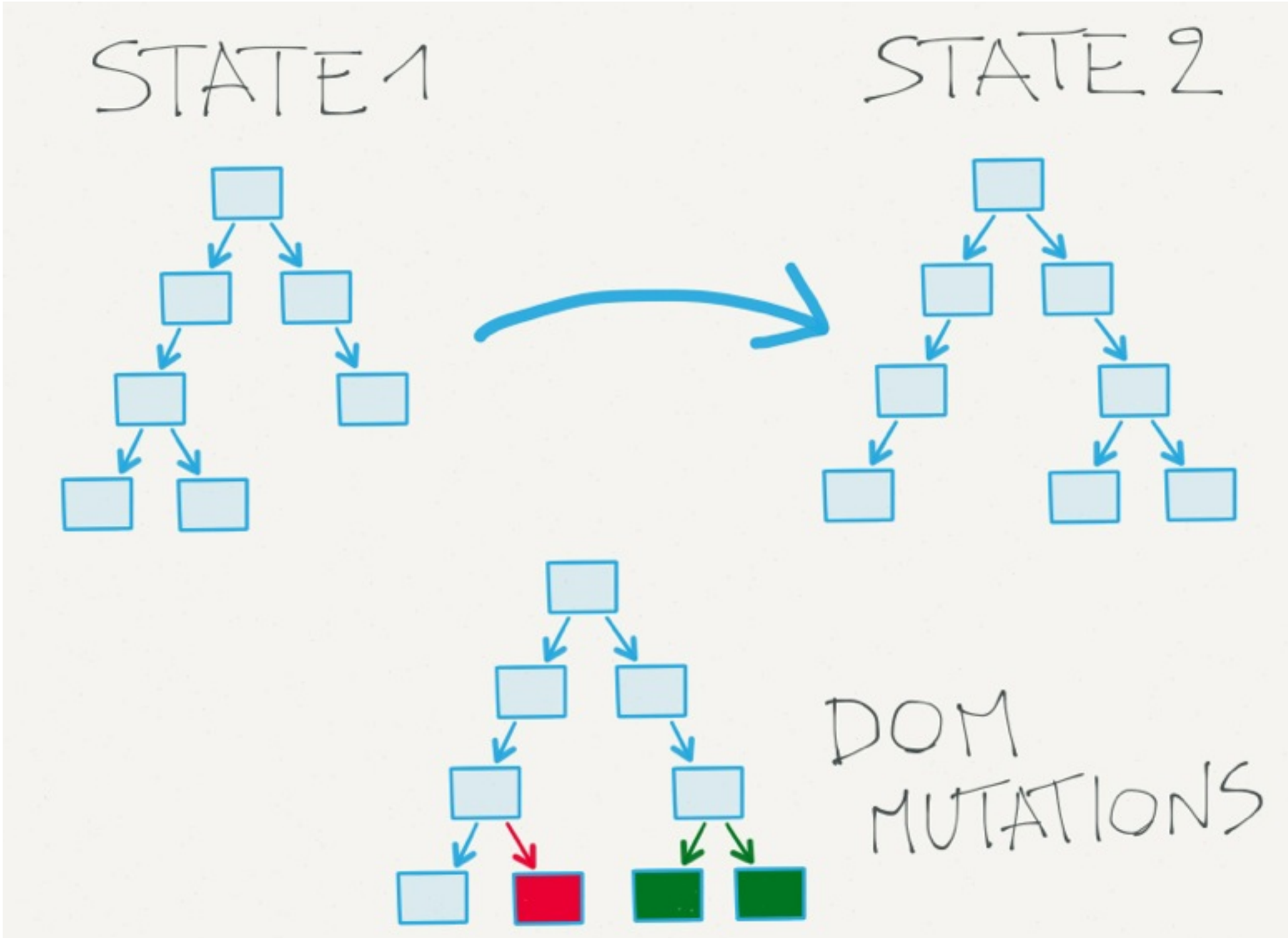
EVERY TIME *setState()*

RE-RENDER *everything*

SIMPLER *mental model*

RE-RENDER... *wat?*

VIRTUAL DOM to the rescue



VDOM facts

VDOM *facts*

RENDER *return* VIRTUAL DOM...

VDOM *facts*

RENDER *return* VIRTUAL DOM...

...AN IN-MEMORY, LIGHTWEIGHT *representation of the* DOM

VDOM facts

RENDER return VIRTUAL DOM...

...AN IN-MEMORY, LIGHTWEIGHT representation of the DOM

REACT is write-only to the DOM

render()

render()

return a

DESCRIPTION

of the UI

introduction to reactjs

STATE → DESCRIBE

[introduction to reactjs]

for every
change of

STATE → DESCRIBE

for every
change of

STATE → DESCRIBE

the whole
user interface

introduction to reactjs

DATA

render(DATA)

render(DATA) → UI

specification

DECLARATIVE, STATELESS APPROACH

over

IMPERATIVE, STATEFUL API

Recap

1. EVERYTHING IS A COMPONENT
2. HOW DATA FLOWS
3. THINK IN STATE, DOM WILL FOLLOW
4. VIRTUAL DOM IS AWESOME

CODING time

clone and play!

react playground on github

discover react example on github

Recap

1. EVERYTHING IS A COMPONENT
2. HOW DATA FLOWS
3. THINK IN STATE, DOM WILL FOLLOW
4. VIRTUAL DOM IS AWESOME

Wait there's more

ISOMORPHIC APP

FLUX

RELAY-GRAPHQL

BAOBAB

CSS IN JS

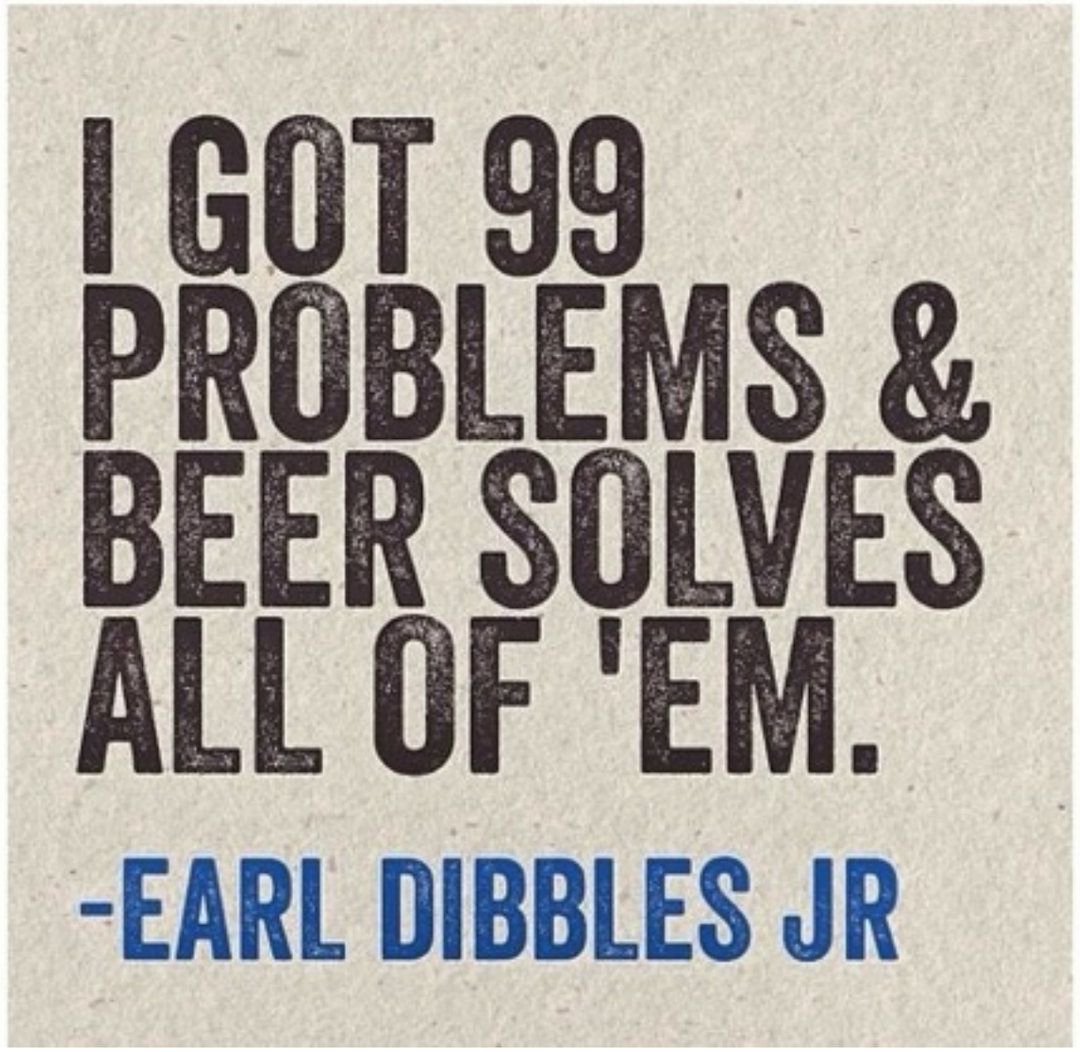
IMMUTABLE-JS

That's all

THANKS!


BEEEEER


BEEEEER!!!


I GOT 99
PROBLEMS &
BEER SOLVES
ALL OF 'EM.
-EARL DIBBLES JR


BEEEEER


WJ