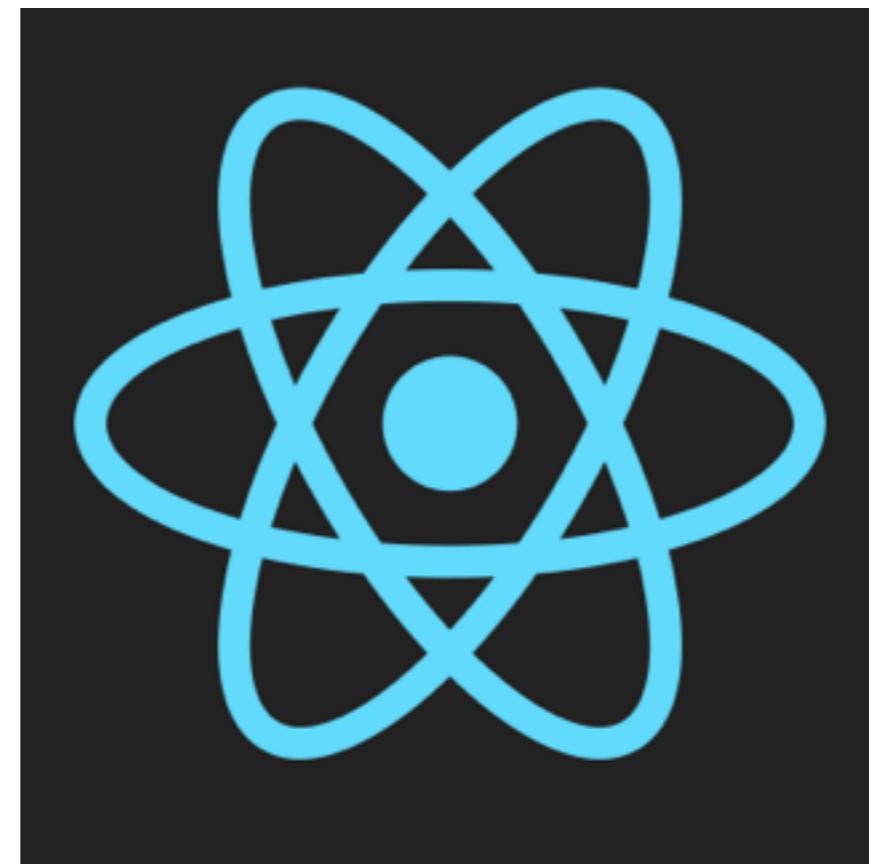


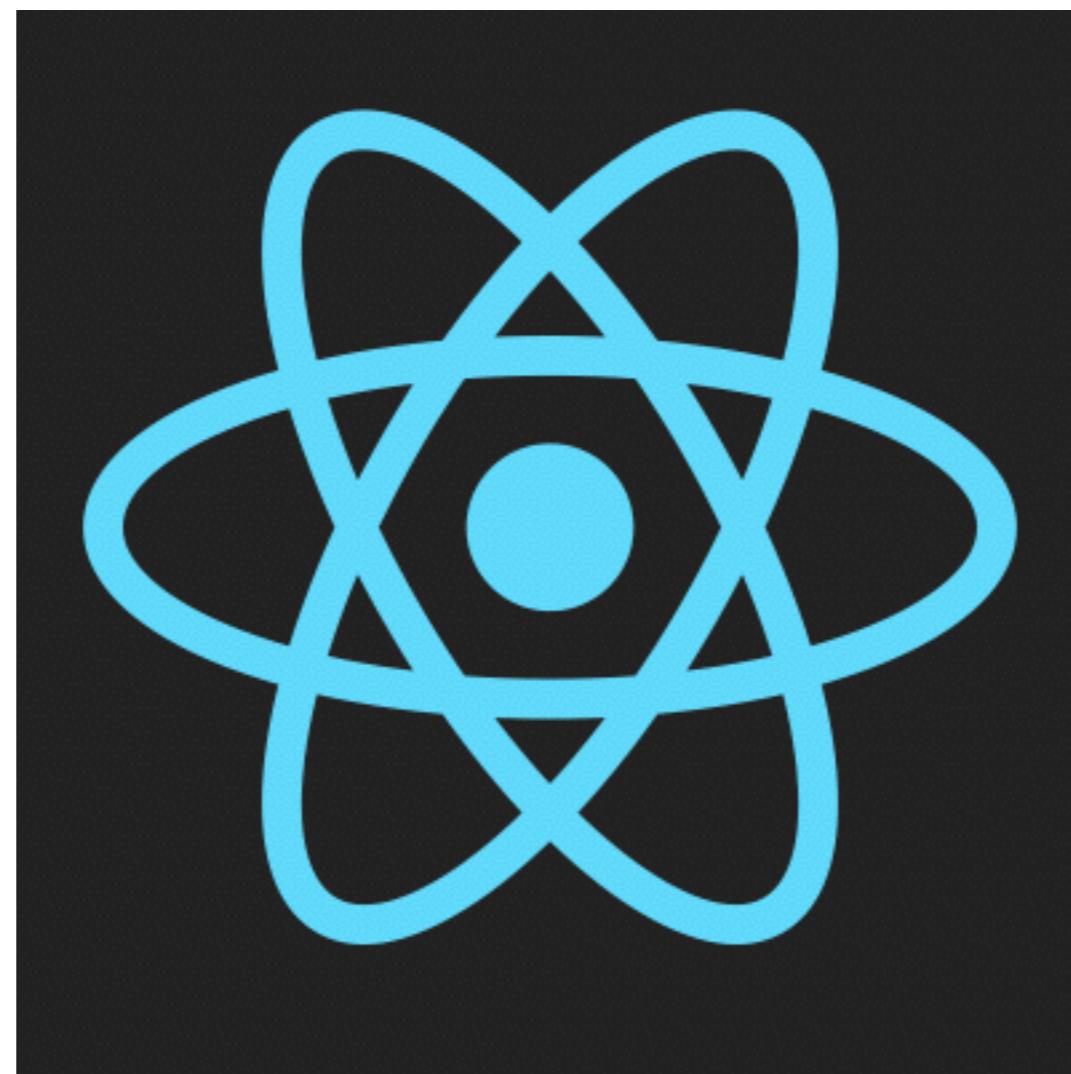
React



Üstün Özgür
@ustunozgur
ustunozgur.com

React.js

User Interface Library



History



JSConf May 2013

Open source



Ben Alman
@cowboy



Following

Facebook: Rethink established best practices™



RETWEETS

18

FAVORITES

18



12:40 AM - 30 May 2013

 [-] **masyI** 24 points 1 year ago

 For a split second I had to ask myself if it was april 1st.
It breaks so many conventions that this library is bound to get more hate then love!
I'm all for diversity, but this one sure smells bad.

 [-] **Lokaltog** 85 points 1 year ago

 Is it just me or does this look a bit messy and cumbersome to work with (and yet another syntax to learn)? What would some of the pros be by using this instead of e.g. Angular?



Instagram



WhatsApp



Flipboard



seller**crowd**

Compare Search terms ▾

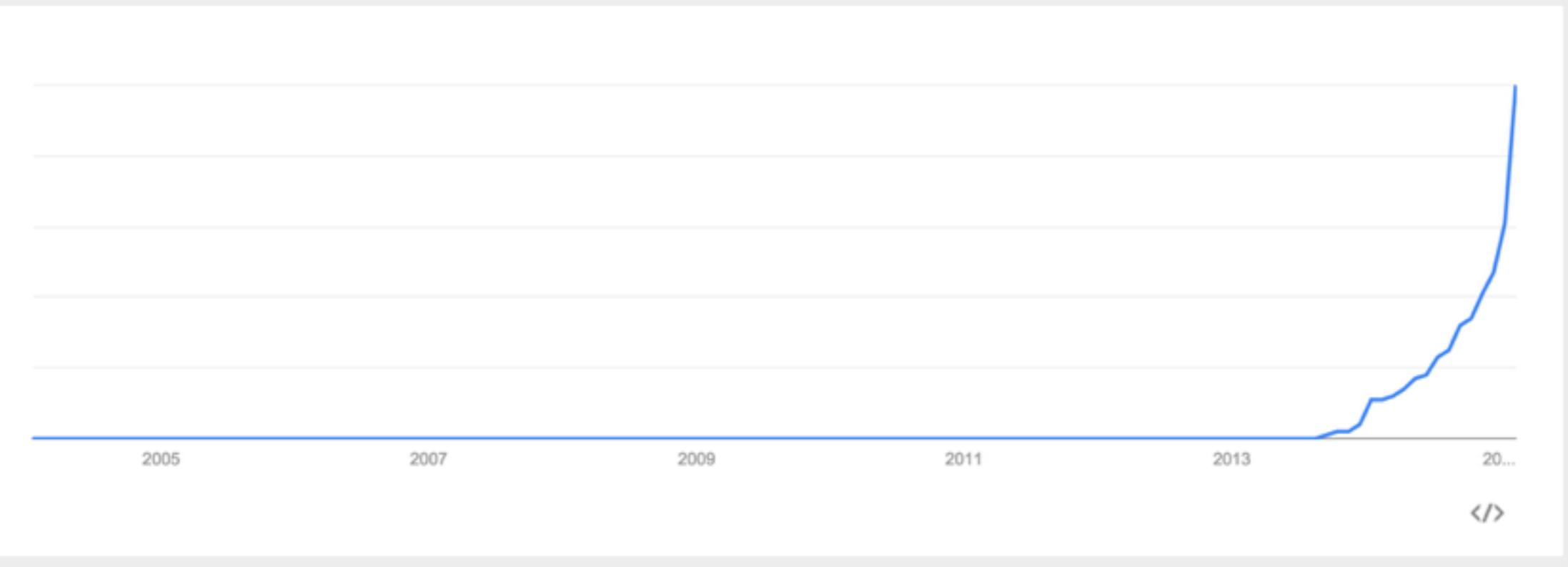
reactjs
Search term

+ Add term

Interest over time ?

News headlines ?

Forecast ?



Compare Search terms ▾

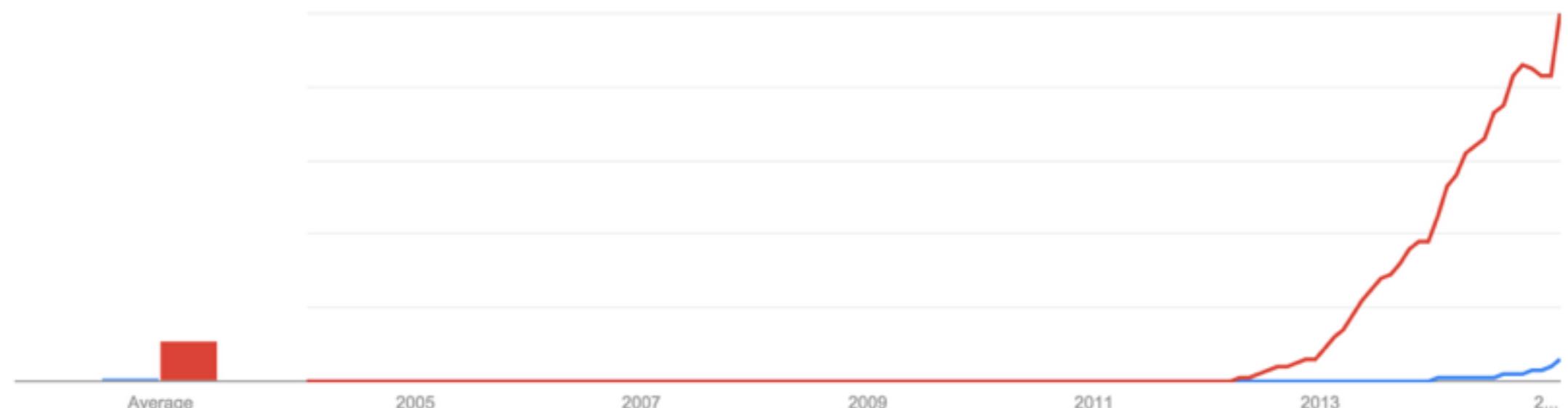
reactjs
Search term

angularjs
Search term

+ Add term

Interest over time ?

News headlines Forecast ?



Average

2005

2007

2009

2011

2013

2014

</>

Compare Search terms ▾

reactjs
Search term

angularjs
Search term

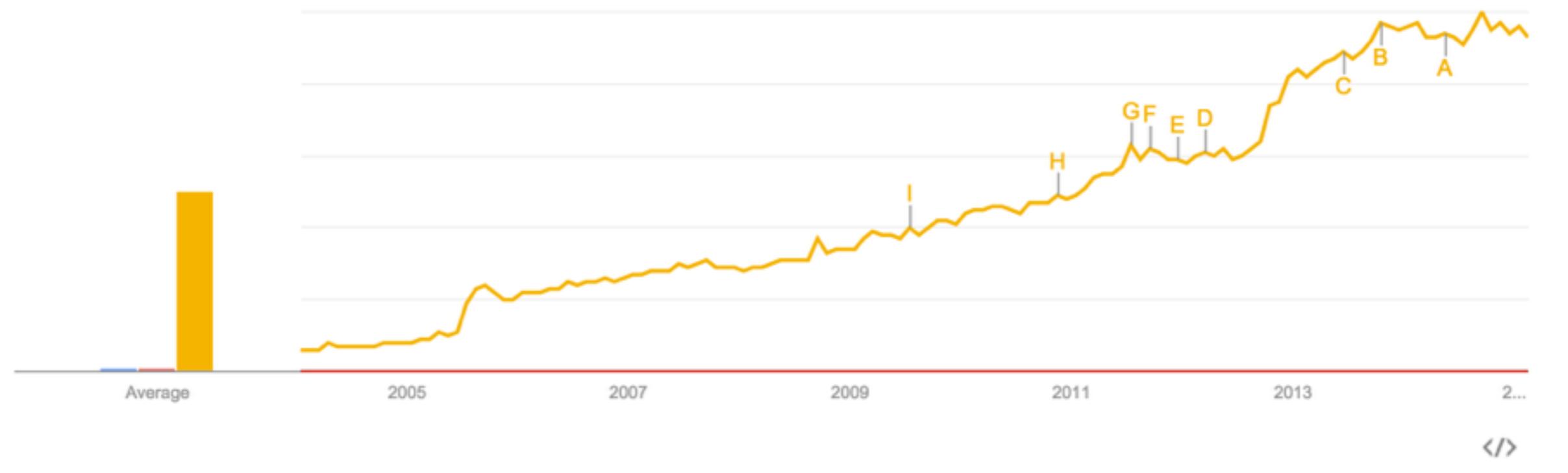
google
Search term

+ Add term

Interest over time



News headlines Forecast



Difficulties in Building User Interfaces

building large applications with
data that **changes over time**, in
a simple and fast manner

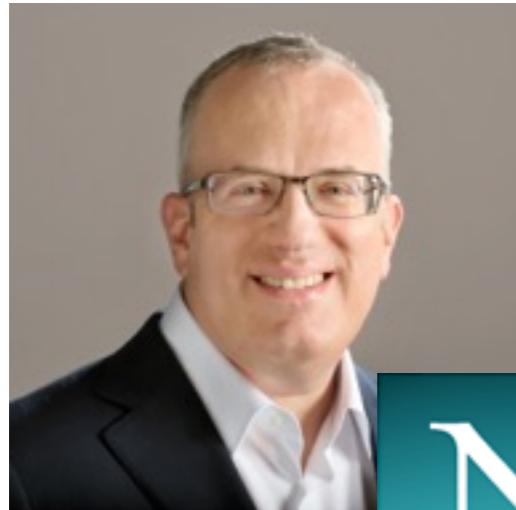
Server Side Interfaces



$f(\text{data}) = \text{html}$

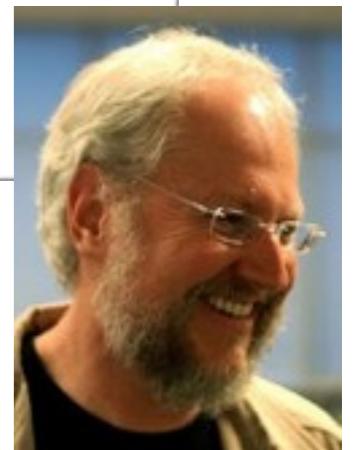
$t=t_0$



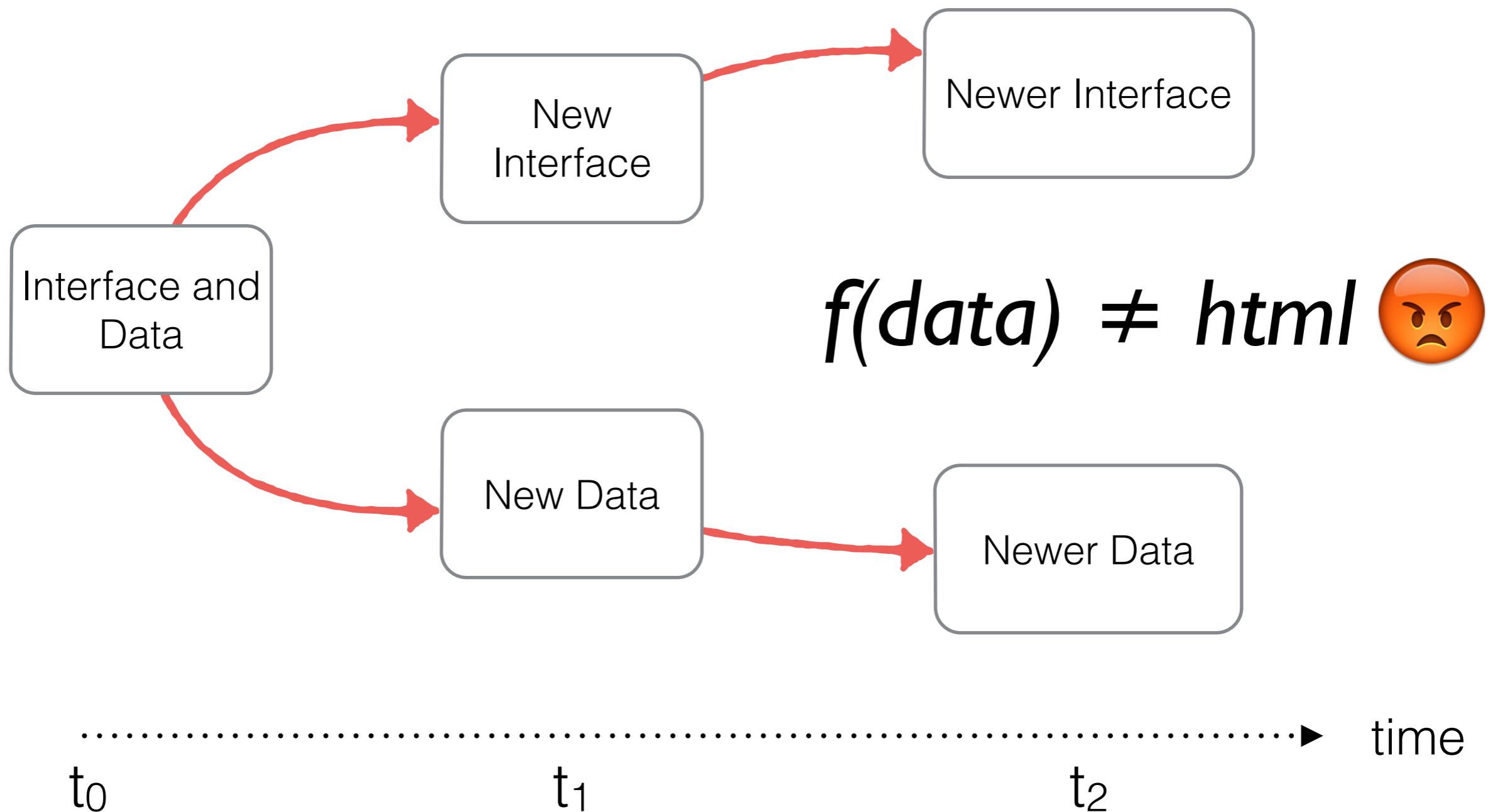


JavaScript

- Developed in 10 days by Eich
- "Lisp in C's clothing" - Crockford
- Very flexible, functional and object-oriented
- CoffeeScript, ES6, underscore...



Client-side interfaces



Example: Chat

10 people online		
	Tolga Duran	Site ●
	Damla Hendriks	Mobil ●
	TC Nihat Özgür	Site ●
	Hasan Tahsin Apakan	
	Vedat Mahir Yilmaz	Site ●
	Ozan Dogu Tuna	36d □
	Celal Can Bilen	Site ●
	Didem Yücel	Site ●
	Sila Karakaya	Mobil ●
	Cicek Sahin	Site ●

Single Data Change
Two View Changes

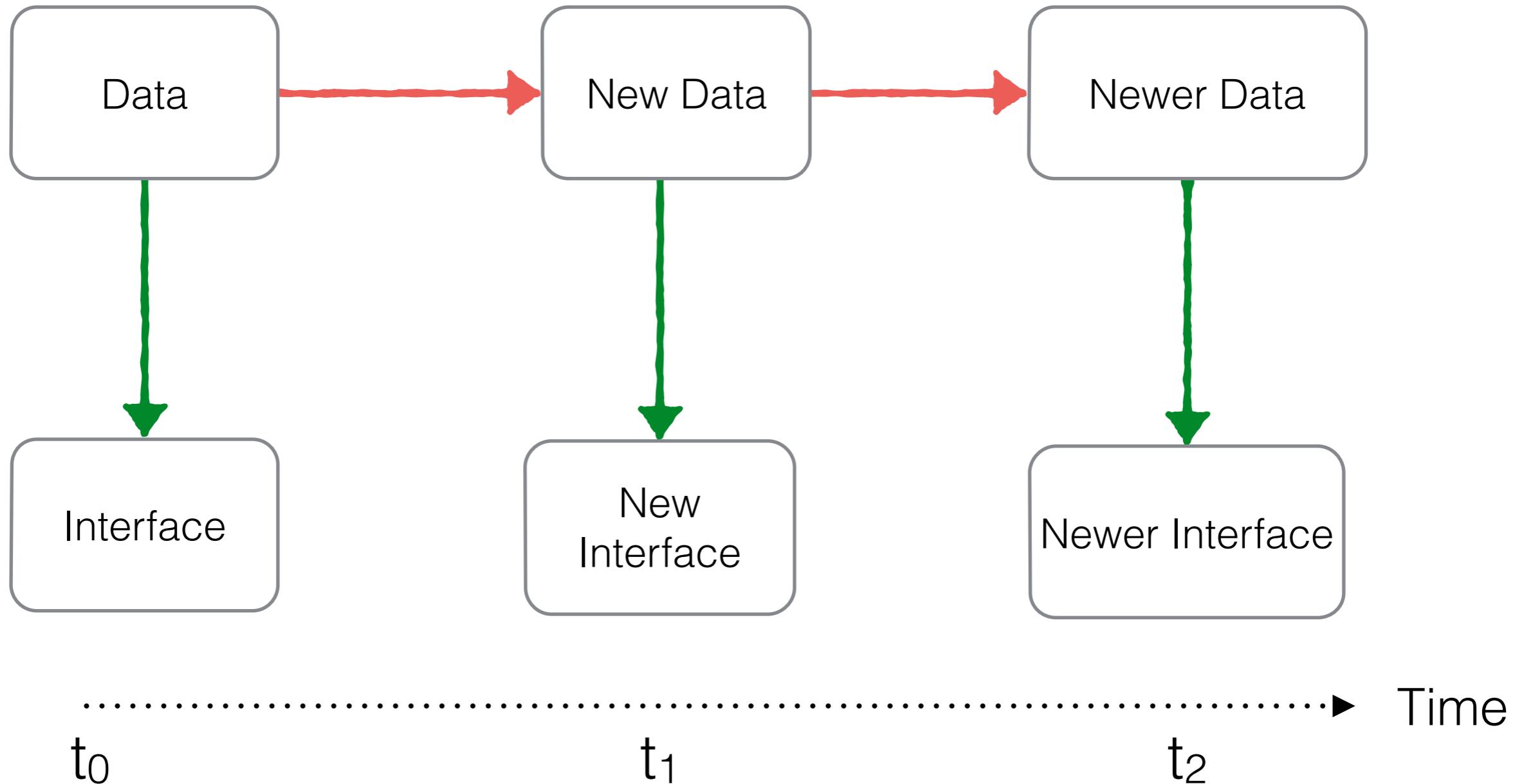
11 people online		
	Tolga Duran	Site ●
	Damla Hendriks	Mobil ●
	TC Nihat Özgür	Site ●
	Hasan Tahsin Apakan	
	Vedat Mahir Yilmaz	Site ●
	Ozan Dogu Tuna	36d □
	Celal Can Bilen	Site ●
	Didem Yücel	Site ●
	Sila Karakaya	Mobil ●
	Cicek Sahin	Site ●
	Kutay Gülay	Site ●

```
$('#people').append(htmlNewPerson)
$('#number').text($('#number').text + 1)
```

Re-render the view
all the time

JS

New View $f(data) = \text{html}$ New Data

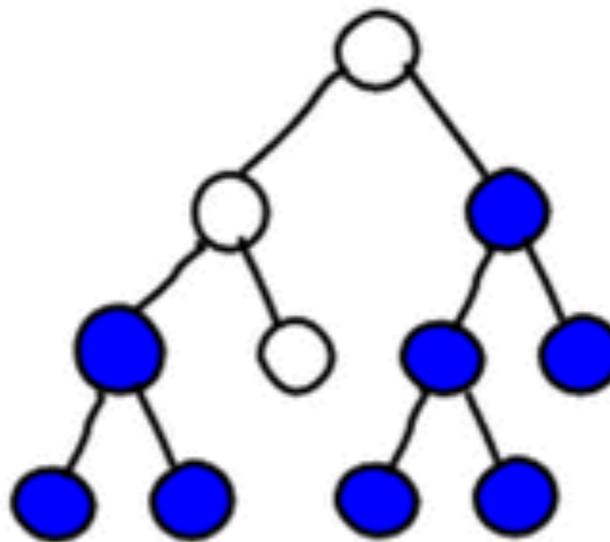


Virtual DOM

Dirty



Re-rendered



Main Principle

Programmer should be focusing
on the data structure behind the
user interface and changes in it
and should define how the data
is going to be viewed just a
single time

Components

Facebook

Ustun Ozgur Profili Düzenle

İnsanları, yerleri ve diğer şeyleri ara

Durum Fotograf/Video Ekle Fotograf Albumu Oluştur

Ne düşünüyorsun?

Karel Barbur, BuzzFeed Video'nun videosunu paylaştı.
1 saat · 

14.697.210 Görüntüleme

BuzzFeed Video
8 Awesome Kids With Awesome Pets
via America's Funniest Home Videos

Beğen · Yorum Yap · Paylaş

4 kişi bunu beğendi.

Pınar Burnukara İlk sahne süper saf mutluluk
1 saat · Beğen · 

Yorum yaz...

Sera Ben buna yorum yaptı.

Ustun Ana Sayfa Arkadaşlarını Bul

Facebook'un React.js..., saat: 19:00
Bilgun Yıldırım'ın doğum günü bugündür

ÖNERİLEN GRUPLAR Tümünü Gör

Ankara Kolejilerin buluşma noktası
Onur Özkoç ve 4 diğer arkadaş katıldı
+ Katıl

Aşk ve Sohbet Odası 39.943 üye
+ Katıl

BU MEMLEKET BİZİM.
TC Nihat Özgür katıldı
+ Katıl

Türkçe · Gizlilik · Kullanıcılar · Çerezler · Reklamlar · Diğer · Facebook © 2015

Deniz Urulbay, Cenk Şaşmazlı'nın gönderisini beğendi.

Karel Barbur kendi gönderisine yorum yaptı.

Elçin Erkin ile Anusha Rajan arkadaş oldu.

Kudret Barlas 

Sevgi Köstel 

Damla Hendriks 39d 

Cağla Tosun İller 

Celal Can Bilen 

Can Akdağ 

Didem Yücel 

Günsu Ozkarar 

Cicek Sahin 

Ozan Dogu Tuna 1s 

Gizem Altay 5s 

Deniz Minican

Kaan Bayka

Hakan Kocaoğlu 8d 

DİĞER ARKADAŞLAR (2)

Mustafa Emre Urfa... 

Search...

Only show products in stock

Name Price

Sporting Goods

Football \$49.99

Baseball \$9.99

Basketball \$29.99

Electronics

iPod Touch \$99.99

iPhone 5 \$399.99

Nexus 7 \$199.99

Search...

Only show products in stock

Name Price

Sporting Goods

Football \$49.99

Baseball \$9.99

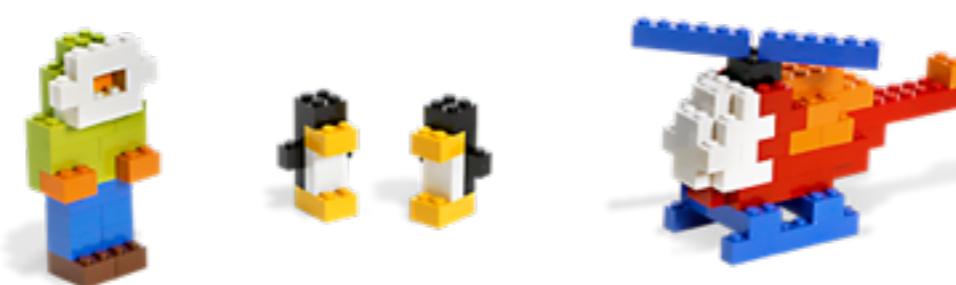
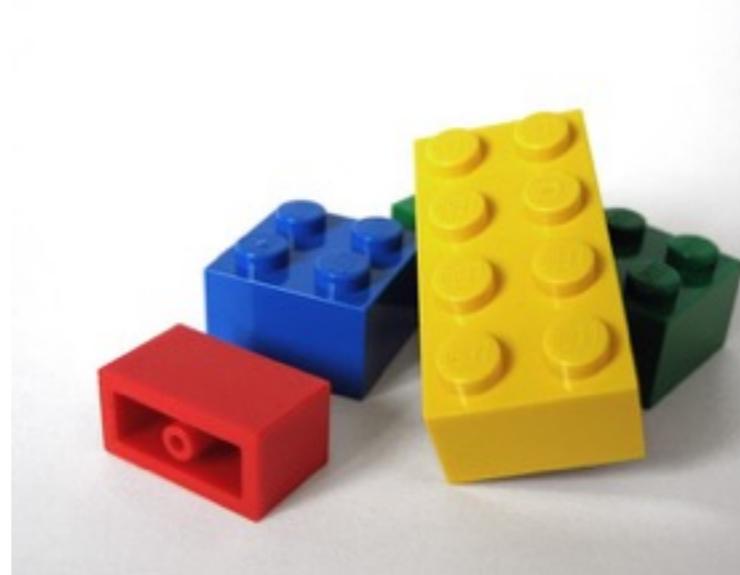
Basketball \$29.99

Electronics

iPod Touch \$99.99

iPhone 5 \$399.99

Nexus 7 \$199.99



A Simple Component



A Simple Component: HelloWorld

```
var HelloWorld = React.createClass({  
  render: function () {  
    return <div>Hello World</div>;  
  }  
})
```

JSX

```
React.createElement("div", null, "Hello World")
```

Mounting on the Page

`React.render(<HelloWorld/>, document.body)`

Component to be Mounted

Existing DOM Node
for Mounting

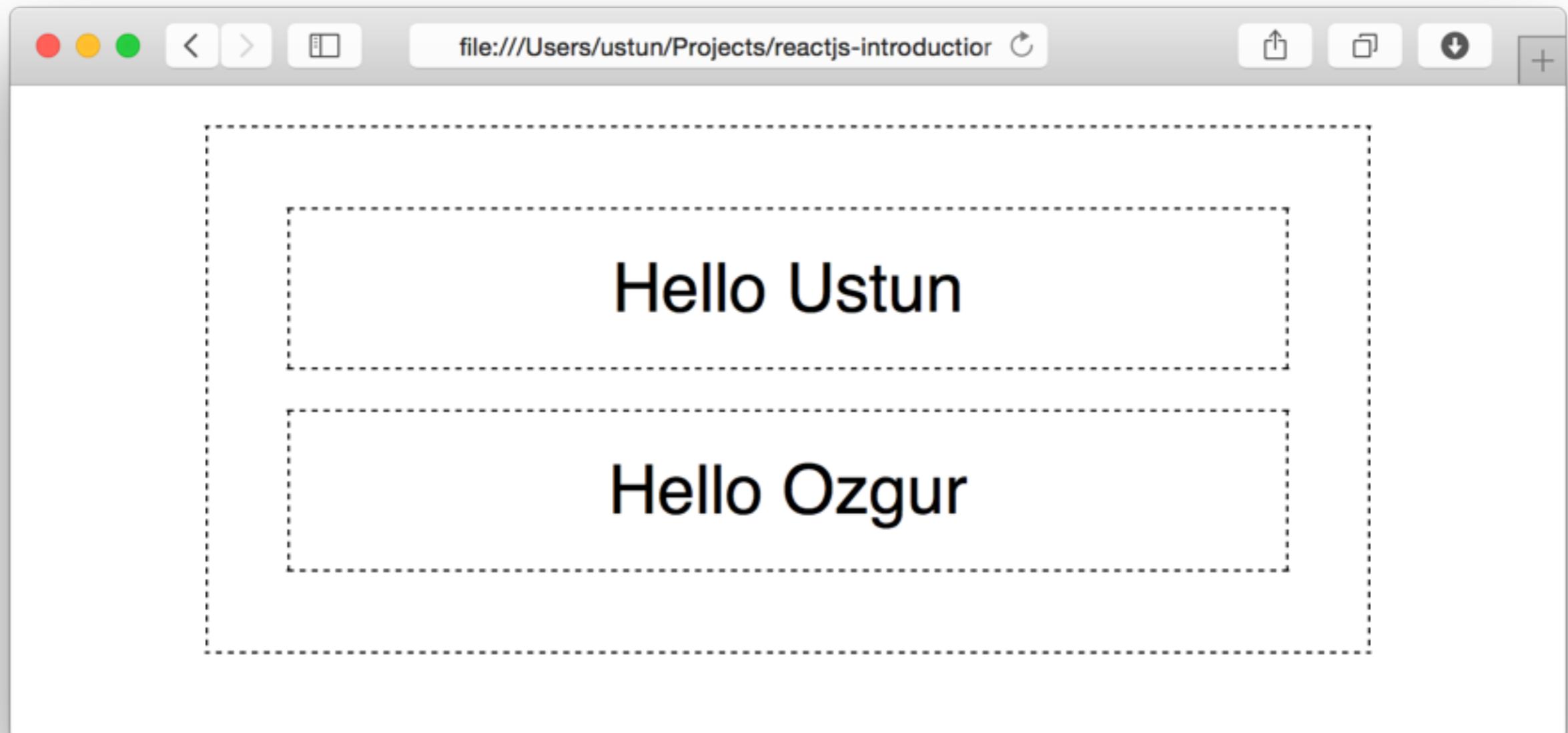
A Compound Component: Greeting



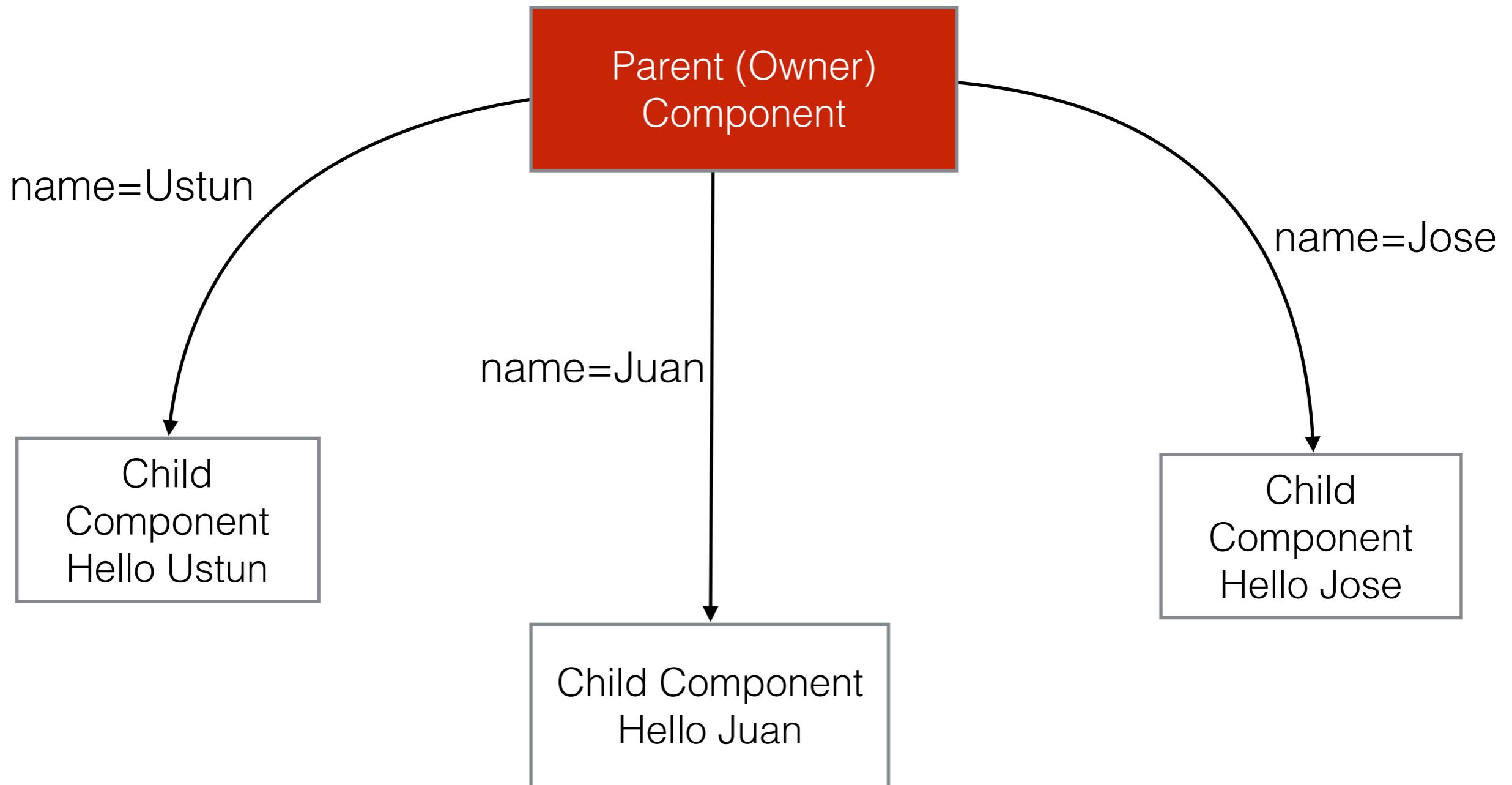
Compound Component: Greeting

```
var Greeting = React.createClass({  
  render: function () {  
    return (  
      <div>  
        <HelloWorld/>  
        <HelloWorld/>  
      </div>)  
  }  
})
```

Parametrized Components



Parametrized Components



HelloWorld with props (properties)

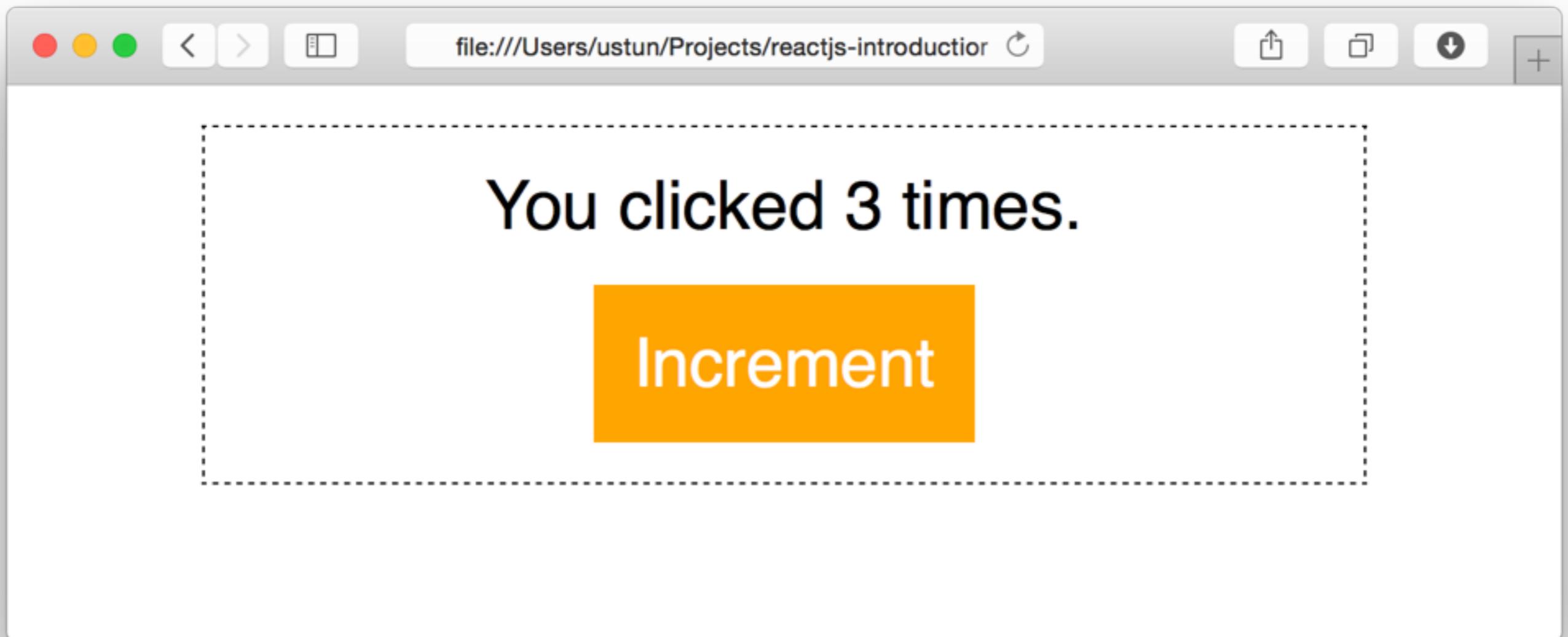
```
var HelloWorld = React.createClass({  
  render: function () {  
    return <div>Hello {this.props.name}</div>  
  }  
})
```

JSX

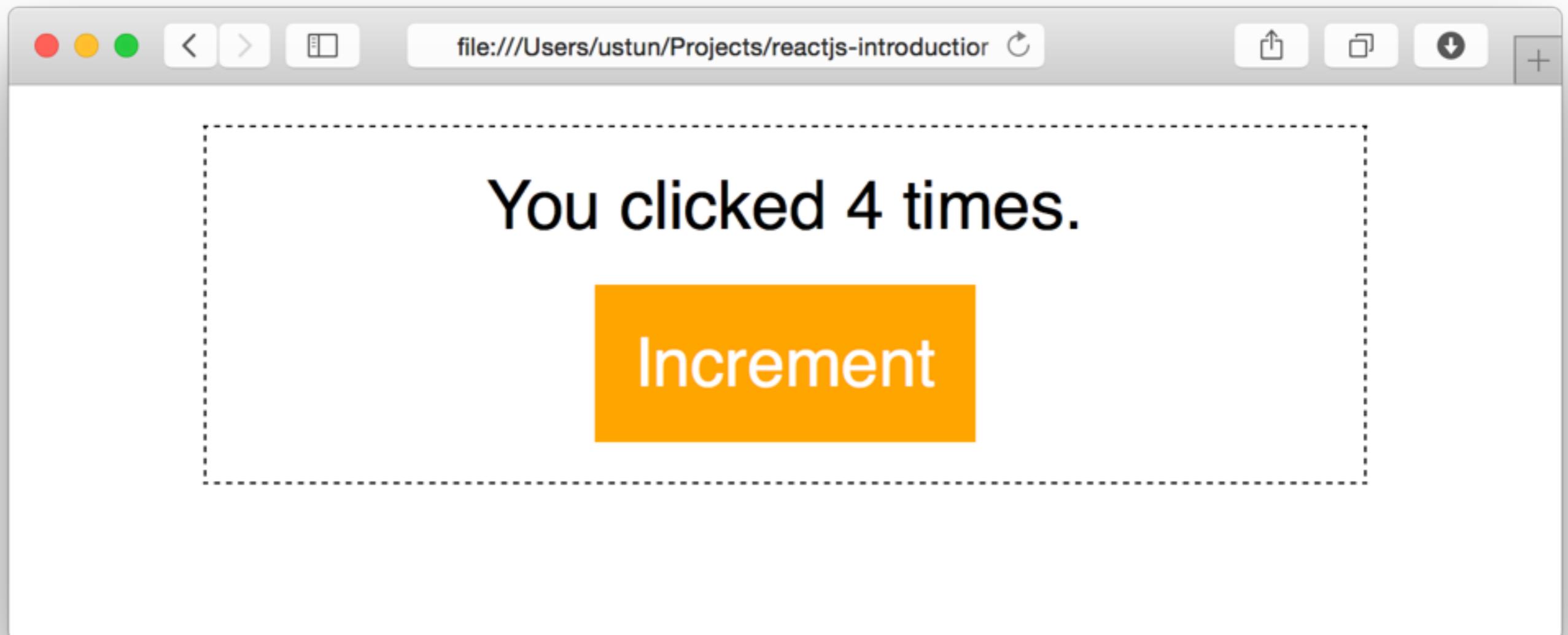
```
React.createElement("div", null,  
  "Hello", this.props.name)
```

```
var Greeting = React.createClass({  
  render: function () {  
    return <div>  
      <HelloWorld name="Üstün"/>  
      <HelloWorld name="Özgür"/>  
    </div>  
  }  
})
```

Mutable Data: *state*



Mutable Data: *state*



```
var Counter = React.createClass({  
  getInitialState: function () {  
    return {counter: 0};  
  },  
  
  render: function () {  
    <div>You clicked {this.state.count} times</div>  
  }  
})
```

```
var Counter = React.createClass({  
  getInitialState: function () {  
    return {counter: 0};  
  },  
  render: function () {  
    return (<div>  
      You clicked {this.state.counter}times.  
      <button>Increment</button>  
    </div>);  
  }  
})
```

```
var Counter = React.createClass({  
  getInitialState: function () {  
    return {counter: 0};  
  },  
  increment: function () {  
    // What should we write here?  
  },  
  render: function () {  
    return (  
      <div>  
        You clicked {this.state.counter} times.  
        <button onClick={this.increment}>Increment</button>  
      </div>;  
    )  
  }  
})
```

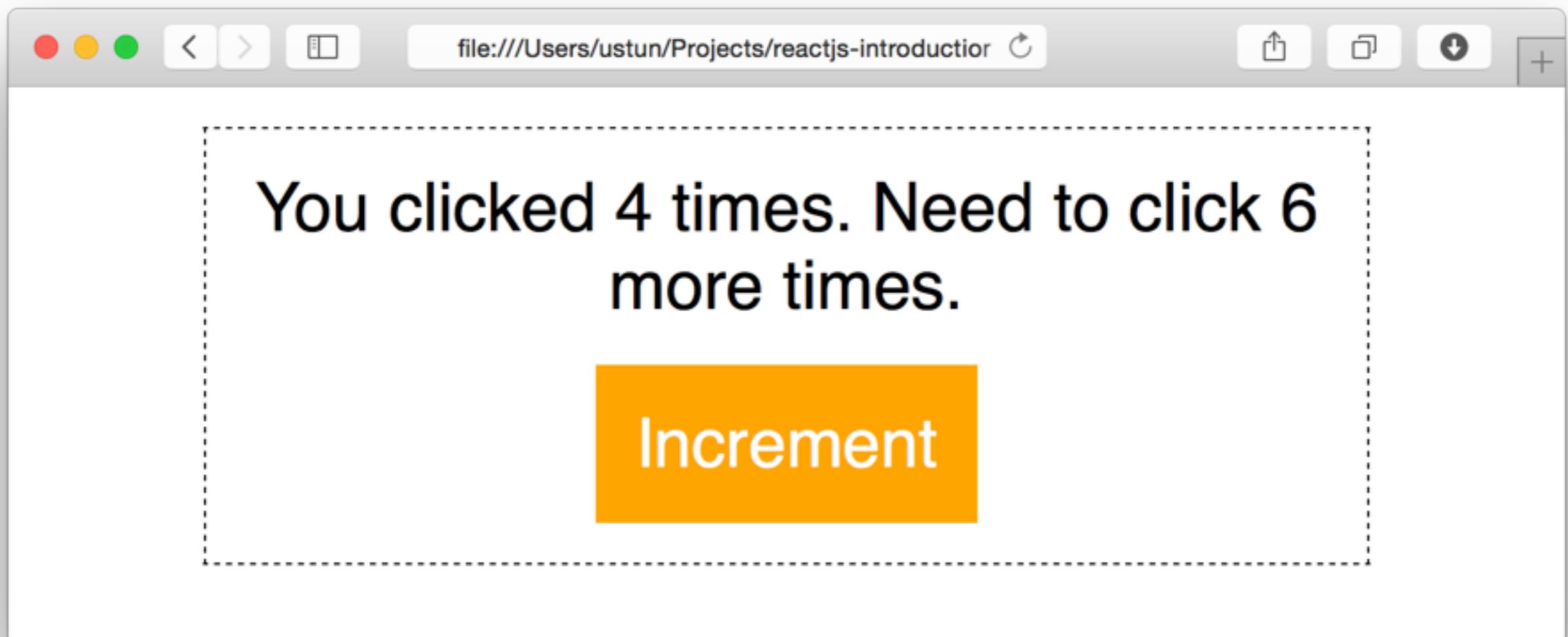
setState!

```
var Counter = React.createClass({  
  getInitialState: function () {  
    return {counter: 0};  
  },  
  increment: function () {  
    var currentCount = this.state.counter;  
    this.setState({counter: currentCount + 1});  
  },  
  render: function () {  
    return (  
      <div>  
        You clicked {this.state.counter} times.  
        <button onClick={this.increment}>Increment</button>  
      </div>)  
    }  
})
```

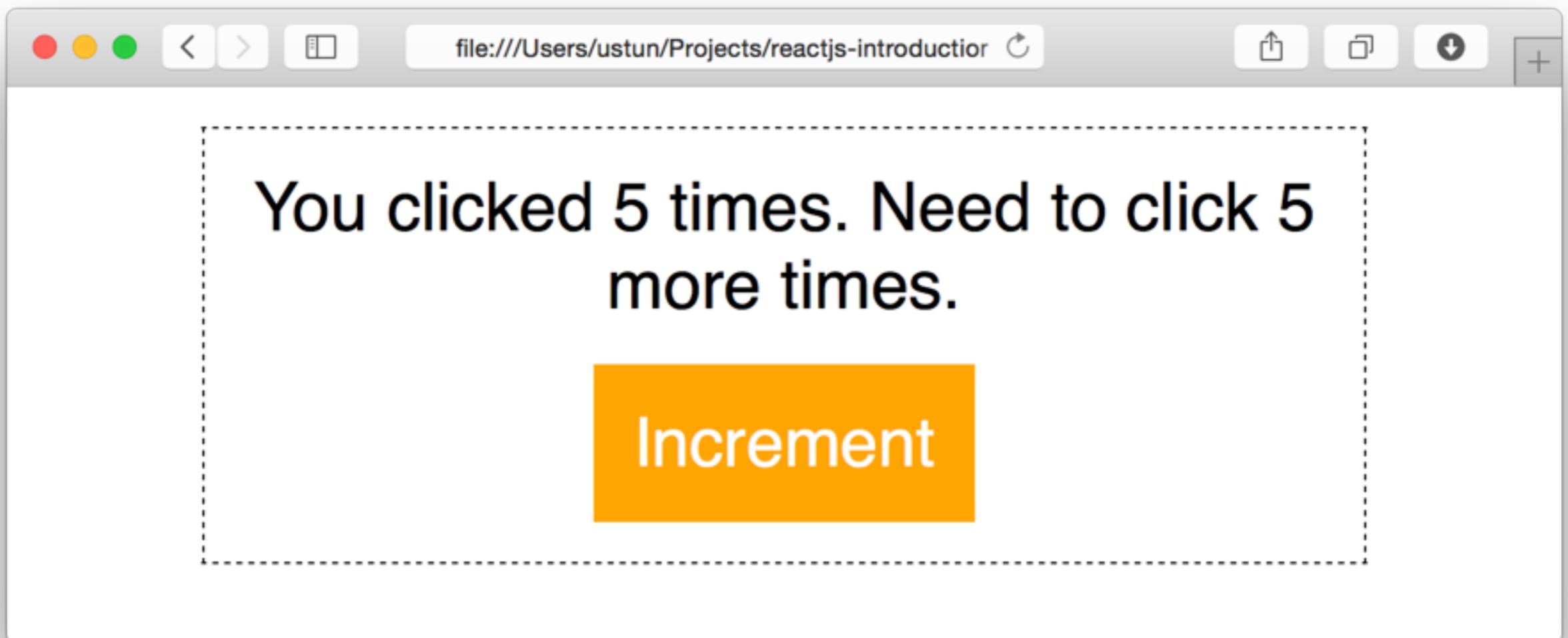
All state changes are explicit

- State change in AngularJS
 - Digest loop
 - Slowness in big applications
 - Might be hard to reason about how the application works

Counter that Counts Forward and Backwards



Counter that Counts Forward and Backwards

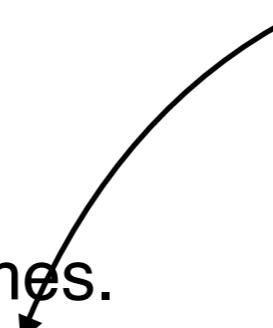


Root data vs. Derived Data

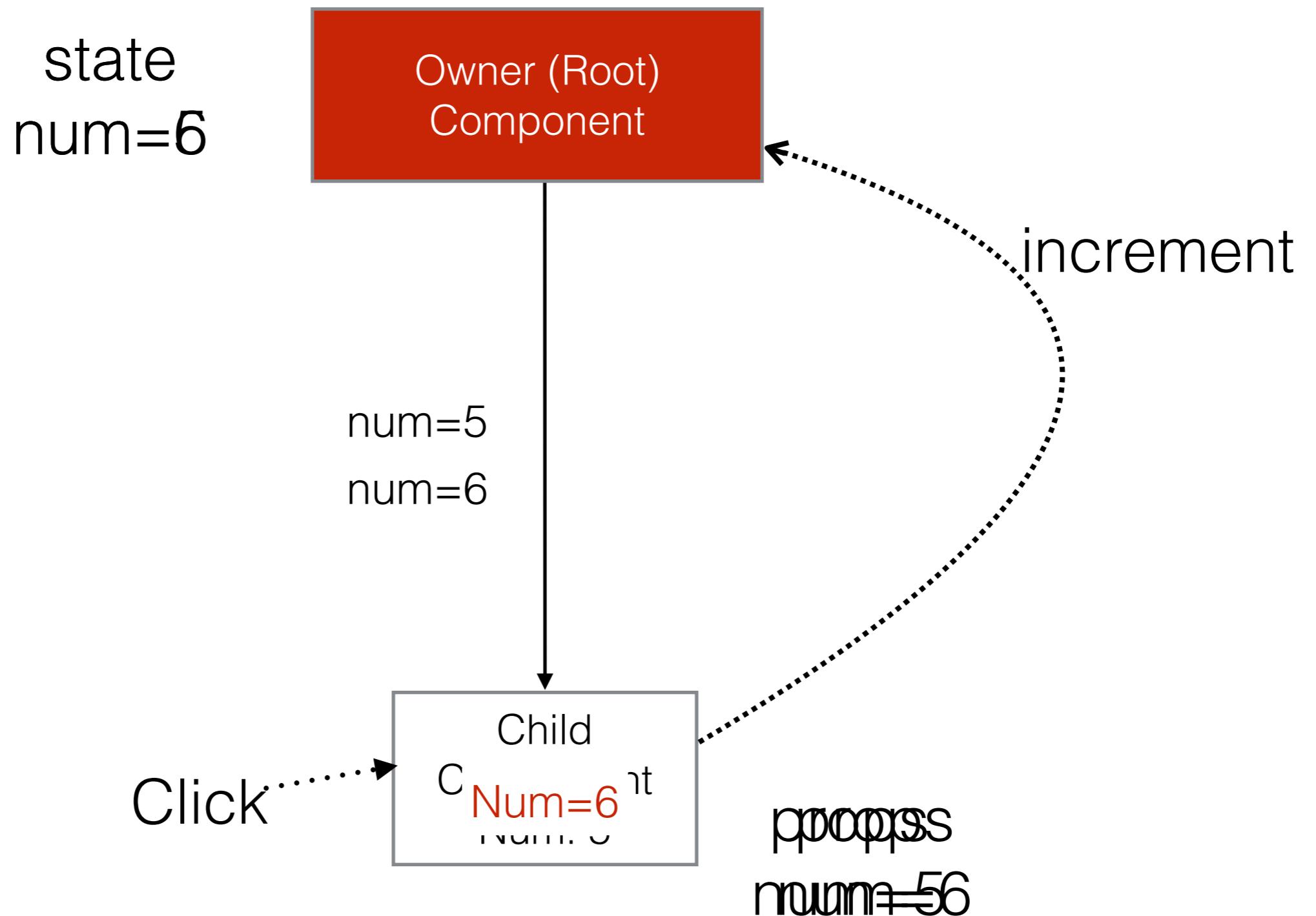
- Single counter value: Single state variable
- Two view values: Derived Data
 - Compute in render method using the state variable

```
var Counter = React.createClass({  
  getInitialState: function () {  
    return {counter: 0};  
  },  
  increment: function () {  
    var currentCount = this.state.counter;  
    this.setState({counter: currentCount + 1});  
  },  
  render: function () {  
    return (  
      <div>  
        You clicked {this.state.counter} times.  
        You have to click {10 - this.state.counter} more times.  
        <button onClick={this.increment}>Increment</button>  
      </div>  
    )  
  }  
})
```

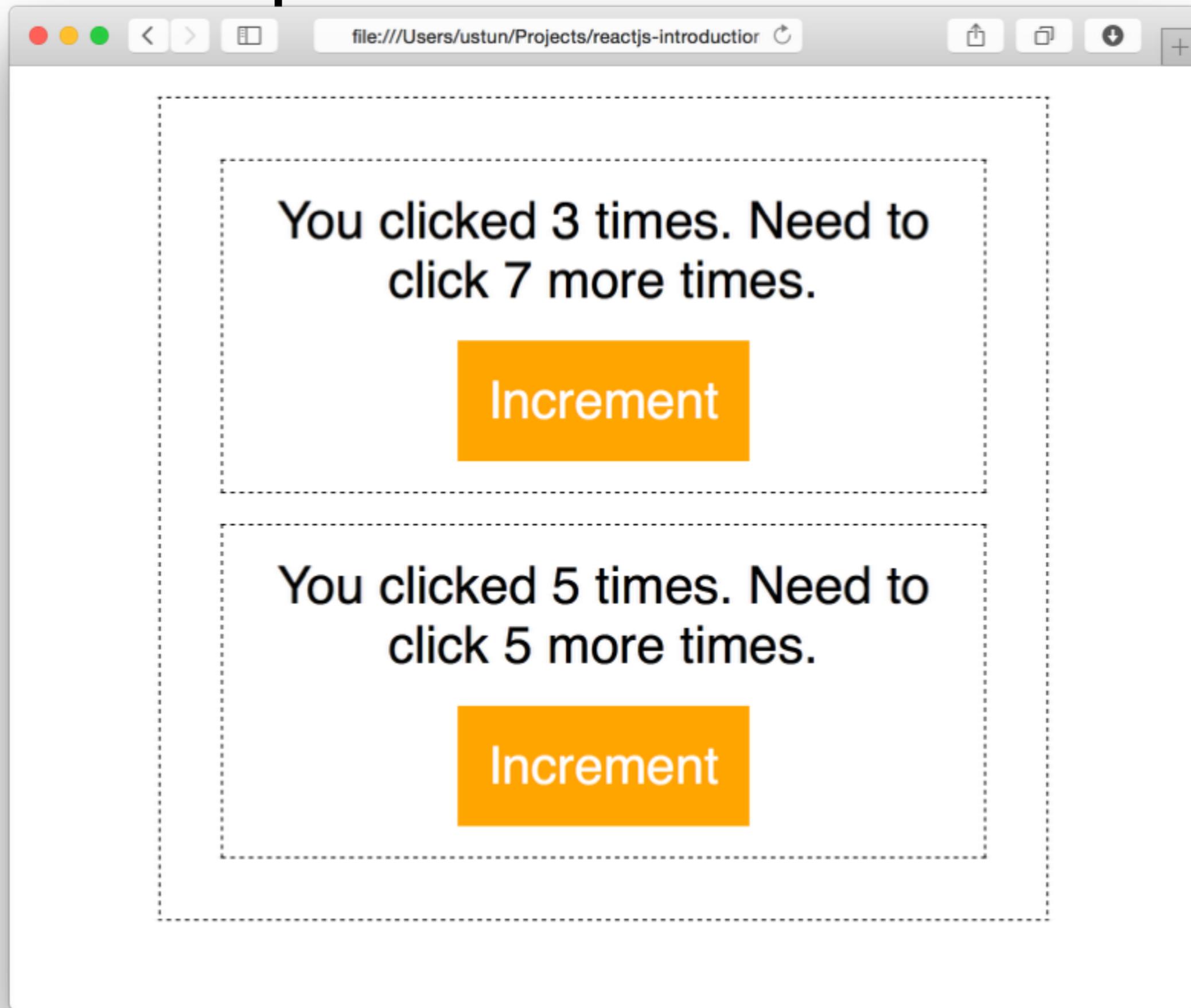
Single Change



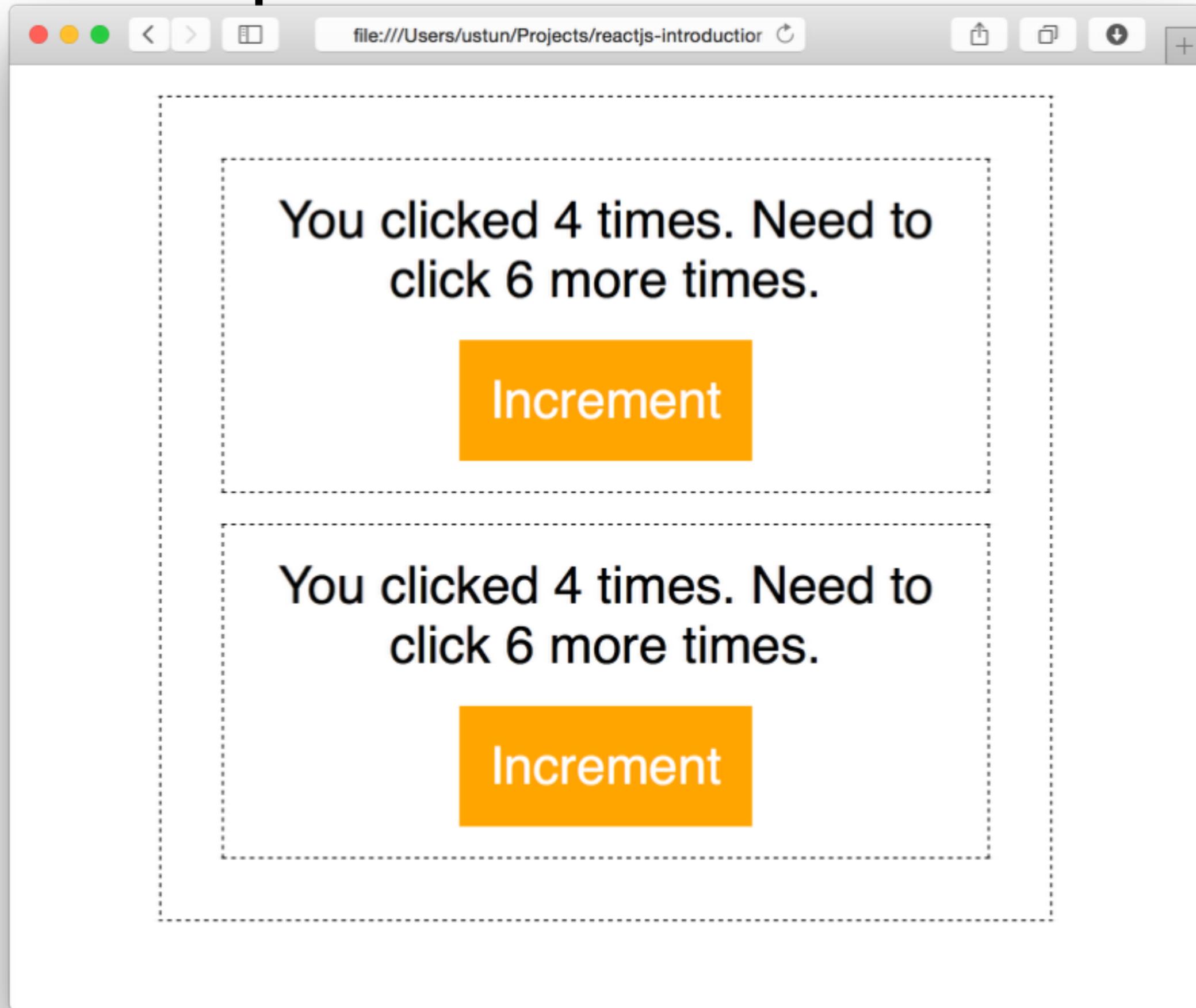
Owner-Child Component Communication



Example: Two Counters



Example: Two Counters

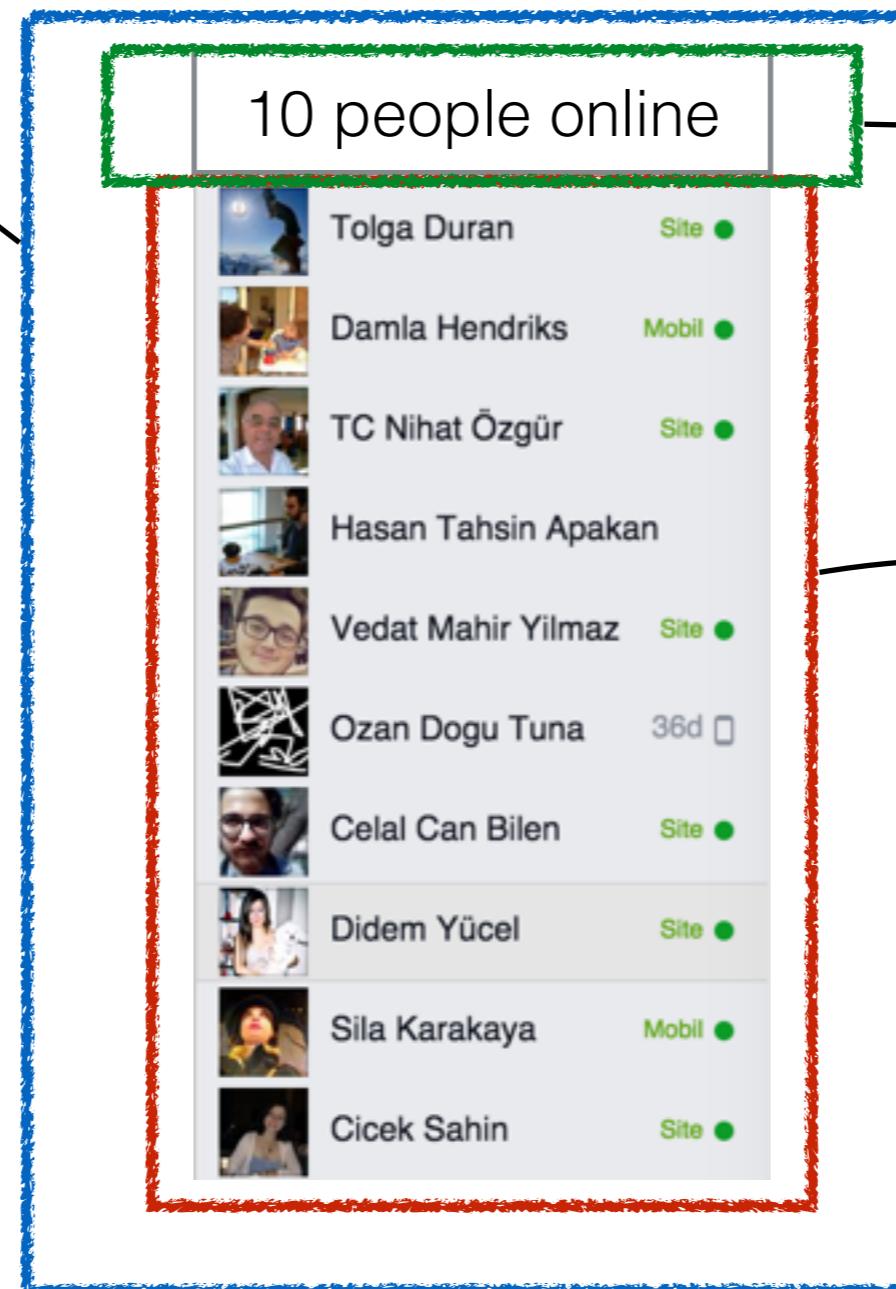


Where Should We Hold State?

- Parent component?
- Child component?

Example: Chat

Chat Component

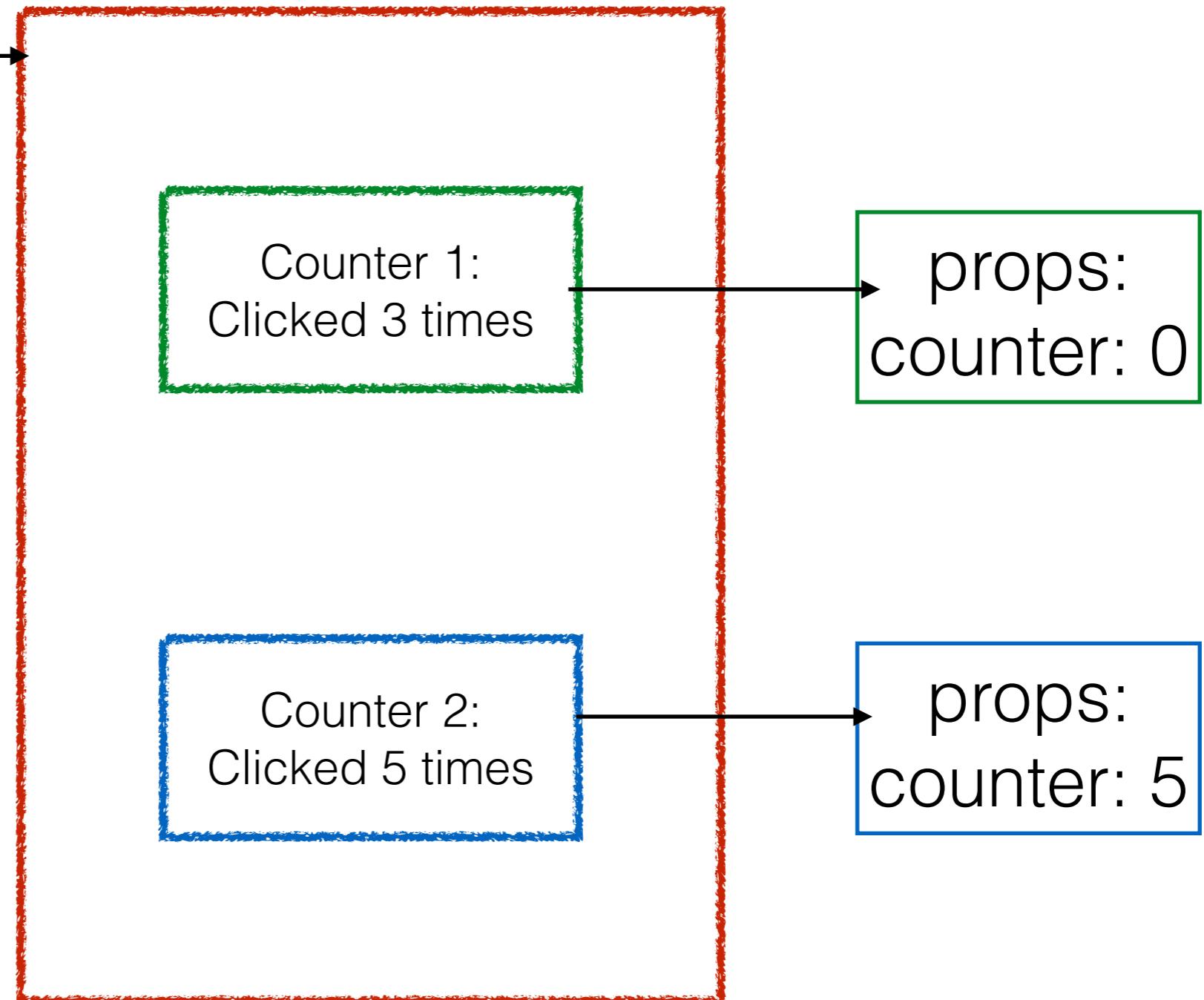


Number Component

Person List Component

state:
counter1: 0
counter2: 5

Counters



```
var Counter = React.createClass({  
  render: function () {  
    return (  
      <div>  
        Clicked {this.props.counter} times  
        Need to click {10 - this.props.counter}  
        more times.  
        <button onClick={this.props.increment}>  
          Increment </button>  
      </div>);  
  }  
})
```

```
var Counters = React.createClass({  
  getInitialState: function () {  
    return {counter1: 3, counter2: 5};  
  },  
  incrementCounter1: function () {  
    this.setState({counter1: this.state.counter1 + 1,  
                 counter2: this.state.counter2 - 1});  
  },  
  incrementCounter2: function () {  
    this.setState({counter1: this.state.counter1 - 1,  
                 counter2: this.state.counter2 + 1});  
  },  
  render: function () {  
    return (  
      <div>  
        <Counter counter={this.state.counter1}  
              increment={this.incrementCounter1}/>  
  
        <Counter counter={this.state.counter2}  
              increment={this.incrementCounter2}/>  
      </div>);  
    }  
  })
```

```
var Counters = React.createClass({  
  getInitialState: function () {  
    return {counter1: 3, counter2: 5};  
  },  
  incrementCounter1: function () {  
    this.setState({counter1: this.state.counter1 + 1,  
      counter2: this.state.counter2 - 1});  
  },  
  incrementCounter2: function () {  
    this.setState({counter1: this.state.counter1 - 1,  
      counter2: this.state.counter2 + 1});  
  },  
  render: function () {  
    return (  
      <div>  
        <Counter counter={this.state.counter1} increment={this.incrementCounter1}/>  
        <Counter counter={this.state.counter2} increment={this.incrementCounter2}/>  
      </div>);  
  }  
})
```

Other Important Topics

- iPhone & Android applications w/ React Native
- server side rendering w/ node.js
- Chrome extension React DevTools
- Facebook's Flux library for big apps

Conclusion

- All state changes are explicit
- Applications divided into Components
- Always re-rendering for data-view synchronization - VirtualDOM

Conclusion

- Repo for today:
- <https://github.com/ustun/react-workshop>
- <https://github.com/ustun/reactjs-introduction>
- Slides and Transcript
- [@ustunozgur](https://twitter.com/ustunozgur)
- ustun@ustunozgur.com