Developer Circles

from **facebook**

Developer Circles

Introduction to ReactJS

Usman Bashir
Developer Circle Lead

facebook for developers

Who am I?

Technology Consultant & Software Developer

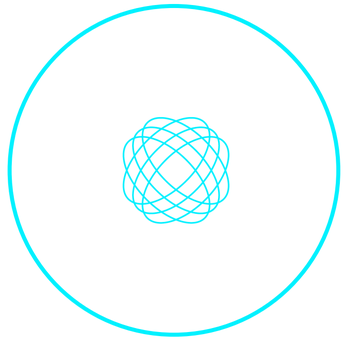
- me@usmanbashir.com
- @ubax
- usmanbashir.com

What is Developer Circles from Facebook?

- Community
- Learn

  - Bots, AI, VR, AR, IoT, React and other technologies

- Collaborate
- Supported by Facebook and local partnerships

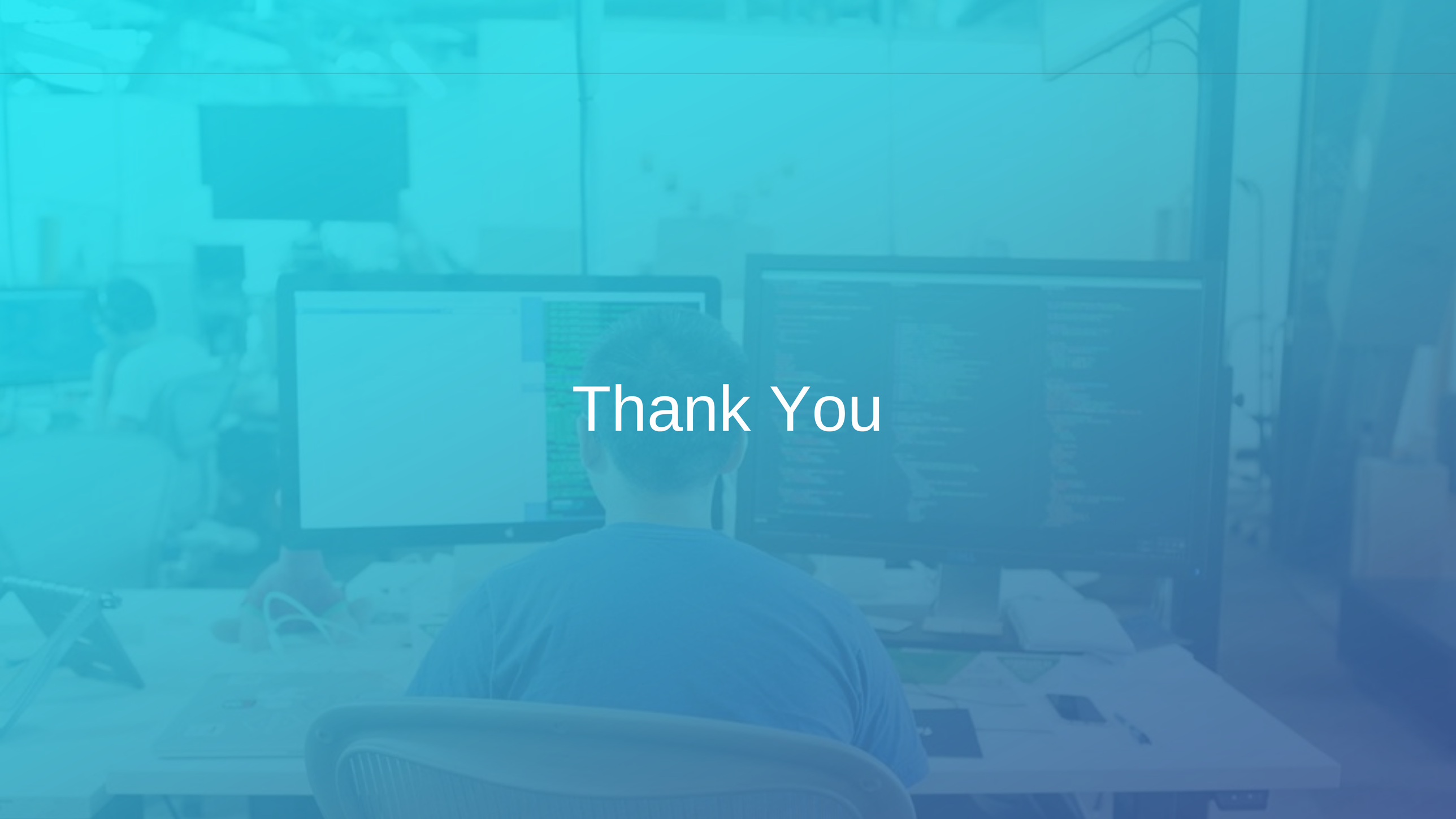


Developer Circle: Jeddah

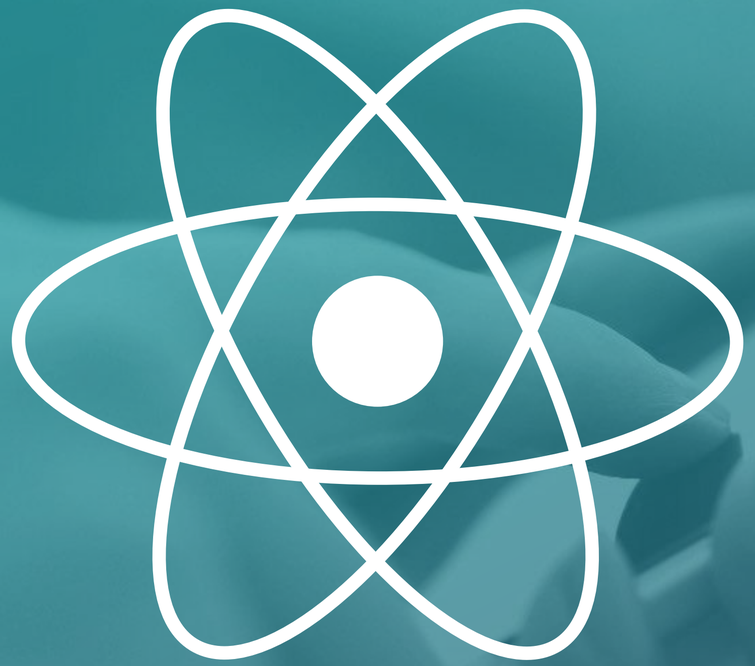
- Tech Talks
- Workshops
- Hackathons
- Viewing Parties
- Hybrid Events

Developer Circle: Jeddah

facebook.com/groups/DevCJeddah



Thank You



Introduction to React

A JavaScript library for building user interfaces

What is React?

One React

To Rule Them All

ReactJS

React Native

React VR

Learn Once,
Write Anywhere


Microsoft

airbnb

WiX.com

Dropbox

SAMSUNG

Pinterest

GitHub

twitter

lyft

stripe

Bloomberg

Baidu 百度

DISCORD

delivery.com

TaskRabbit

KHANACADEMY

Prerequisites

ES6

arrow functions

let

classes

const

template literals

What does React brings to the table?

Components

Virtual DOM

Props & State

Imperative

vs.

Declarative

Example 1

Imperative

(the old way)

```
funtion submitForm() {
  // …

  var submitButton = $(‘button-id’);

  if (isFormValid === false) {
    submitButton.prop('disabled', true)
                .addClass('disabled');
  } else {
     submitButton.prop('disabled', false)
                 .removeClass('disabled');
  }

  // …
}
```

Declarative

(the new way)

```
<SubmitButton
  disabled={this.state.isFormValid} />
```

Example 2

Imperative

(the old way)

```
if (userLikes()) {
  if (!hasBlueLike()) {
    removeGrayLike();
    addBlueLike();
  }
} else {
  if (hasBlueLike()) {
    removeBlueLike();
    addGrayLike();
  }
}
```

Declarative

(the new way)

```
if (this.state.liked) {
  return (<BlueLike/>);
} else {
  return (<GrayLike/>);
}
```

UI = RENDER(state)

Components

Declarative Components

```
<Button />
```

```
class Button extends React.Component {
 render() {
   return <input type="button" value="I'm a button!" />;
  }
}
```

```
class Button extends React.Component {
  render() {
    return <input type="button" value="I'm a button!" />;
  }
}
```

```
class Button extends React.Component {
 render() {
    return <input type=“button” value=“I'm a button!” />;
  }
}
```

Composed Components

```
<PlainMenu>
  <Button />
  <Button />
  <Button />
  <Button />
<PlainMenu />
```

```
<Header>
  <Logo />
  <Title />
  <PlainMenu />
<Header/>
```

```
<Header>
  <Logo />
  <Title />
  <PlainMenu />
<Header/>
```

Reusable Components

```
<App>
  <Header />
  <SearchBar />
  <BlogPost />
  <Footer>
    <Copyright />
    <PlainMenu />
  <Footer />
<App />
```

```
<App>
  <Header />
  <SearchBar />
  <BlogPost />
  <Footer>
    <Copyright />
    <PlainMenu />
  <Footer />
<App />
```

UI composed of
<components />

Fast Rendering With Virtual DOM

React Virtual DOM

<h1>Before<h1/>

Virtual DOM Before

<h1>After<h1/>

Virtual DOM After

React Virtual DOM

PATCH

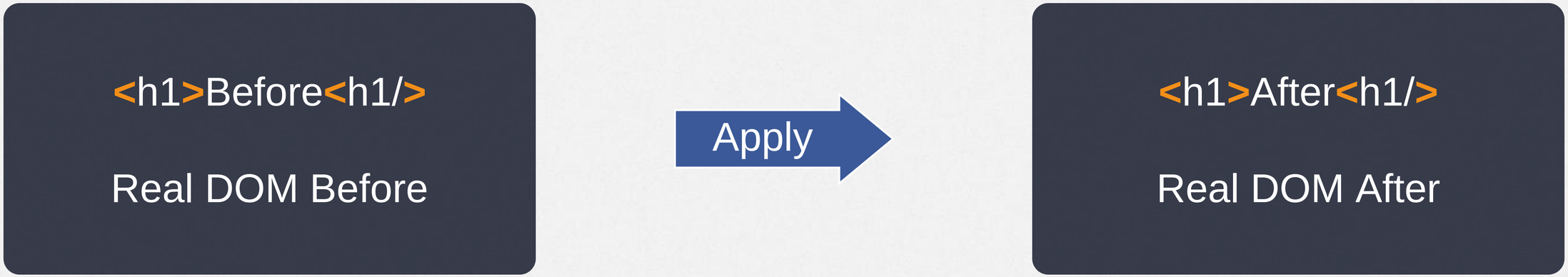
...

Set <h1> to After

...

Batch of DOM operations

Browser DOM



Data Flow

Props

```
this.props
```

```
<Welcome name='Edward Elric' />
```

```
class Welcome extends React.Component {
 render() {
    return <h1>Hello, {this.props.name}!</h1>;
  }
}
```

```
class Welcome extends React.Component {
 render() {
    return <h1>Hello, {this.props.name}!</h1>;
  }
}
```

Pure Functions

Data Flow State

```
this.state
```

Imperative vs. Declarative

After Credits

Declarative

(the new way)

```
<SubmitButton
  disabled={this.state.isFormValid} />
```

```
class SubmitButton extends React.Component {
 render() {
    const classes = `button submit ${this.props.disabled ? “disabled”  : “”}`;

    return <input type='button‘
                  value='Submit‘
                  disabled={this.props.disabled}
                  className={classes} />;
  }
}
```

```
class SubmitButton extends React.Component {
 render() {
    const classes = `button submit ${this.props.disabled ? “disabled” : “”}`;

    return <input type='button‘
                  value='Submit‘
                  disabled={this.props.disabled}
                  className={classes} />;
  }
}
```

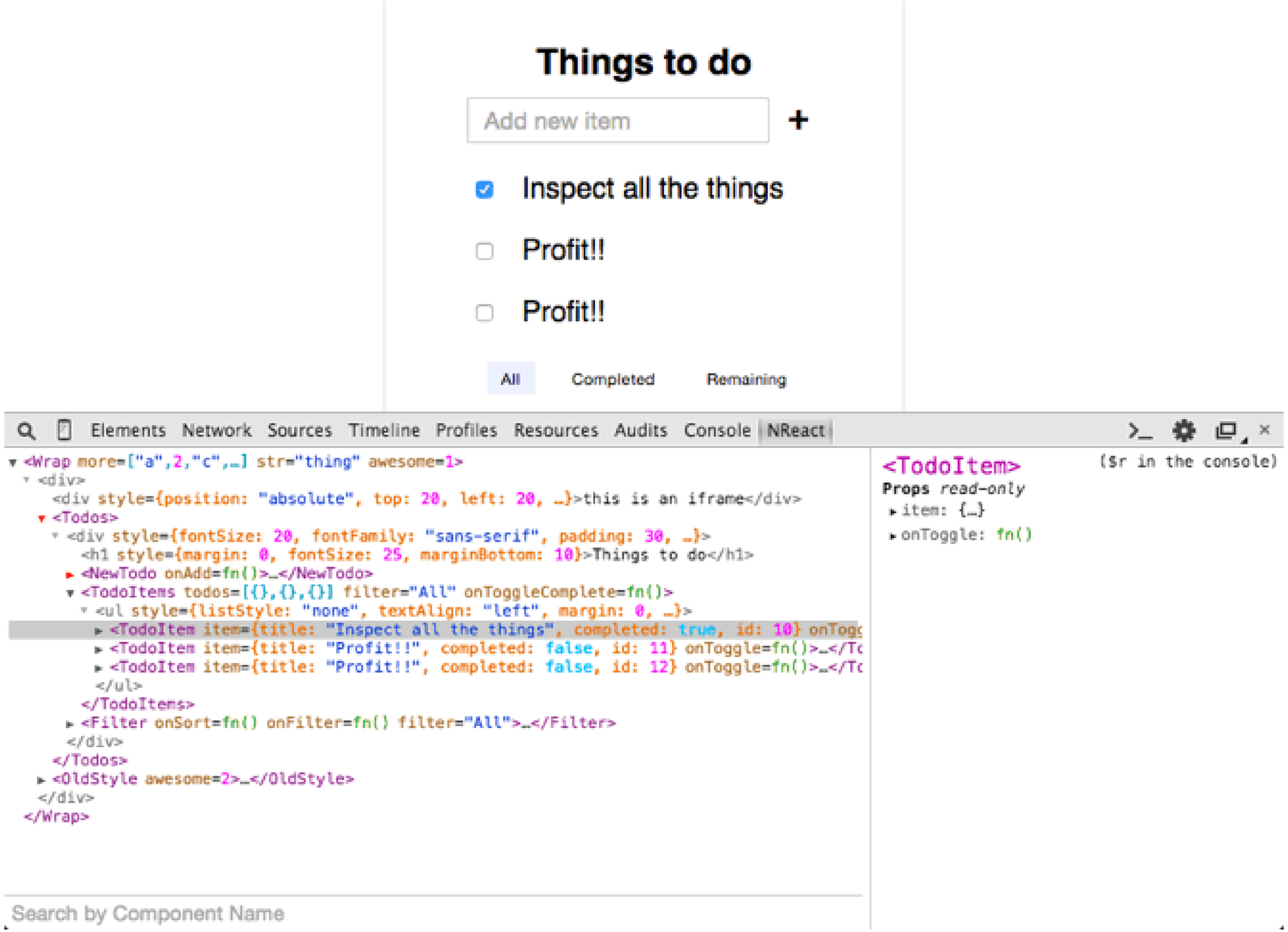
```
const classes = `button submit ${this.props.disabled ? “disabled”  : “”}`
```

Template literal

Developer Tools

Create React App

React Developer Tools


Things to do
Add new item
+
Inspect all the things
Profit!!
Profit!!
All
Completed
Remaining
Elements Network Sources Timeline Profiles Resources Audits Console NReact
<Wrap more=["a",2,"c",…] str="thing" awesome=1>

<div style={position: "absolute", top: 20, left: 20, …}>this is an iframe
<Todos>
<div style={fontSize: 20, fontFamily: "sans-serif", padding: 30, …}>
<h1 style={margin: 0, fontSize: 25, marginBottom: 10}>Things to do</h1>
<NewTodo onAdd=fn()>…</NewTodo>
<TodoItems todos=[{},{},{}] filter="All" onToggleComplete=fn()>
<ul style={listStyle: "none", textAlign: "left", margin: 0, …}>
<TodoItem item={title: "Inspect all the things", completed: true, id: 10} onTogg
<TodoItem item={title: "Profit!!", completed: false, id: 11} onToggle=fn()>…</Tc
<TodoItem item={title: "Profit!!", completed: false, id: 12} onToggle=fn()>…</Tc

</TodoItems>
<Filter onSort=fn() onFilter=fn() filter="All">…</Filter>

</Todos>
<OldStyle awesome=2>…</OldStyle>

</Wrap>
<TodoItem>
($r in the console)
Props read-only
item: {…}
onToggle: fn()
Search by Component Name

Developer Circles

Workshop

Usman Bashir
Developer Circle Lead

facebook for developers

What's Next?

- Notes & Resources
- Learn more
- Build something
- Share it with the community

Thank You

For real, this time!