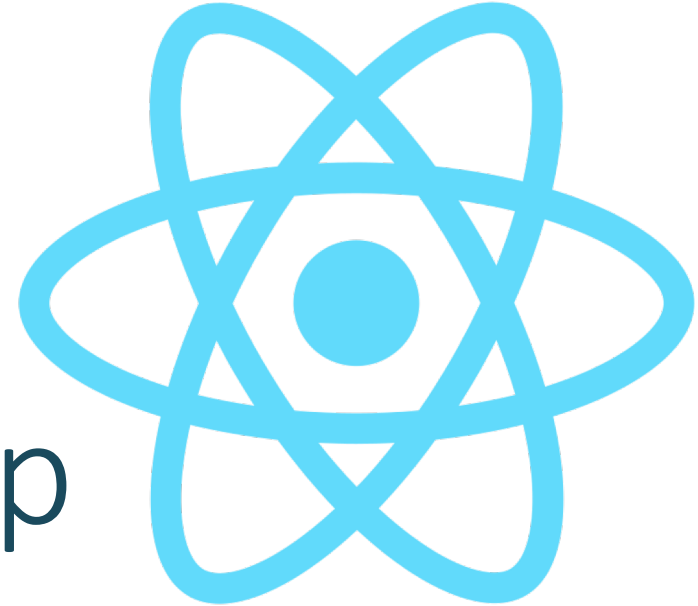
React workshop



APRIL 2018

UVdata at UCN

Wouter van Vliet & Valerie Beltrame

Who are we?

Presentation overview

- About UVdata
- Introduction to React
- Getting started
- Demo
- Build your first components

About UVdata

About UVdata

- Online solutions for educational purposes
- 9 frontend developers working with React daily
- React in 2 projects: UVvej, MinUddannelse
- Ca. 400.000 active users/month
- Default choice for new projects

Technology Stack

- React
- Jade/Pug & jQuery (legacy)
- REST api
- Backend: .NET

Who else uses React?

WordPress

And many more...

Source: https://github.com/facebook/react/wiki/Sites-Using-React

What is React?

What is React?

React is a **JavaScript** view library that renders markup from **reusable components**.

Given the same **props** and **state**, a component always produces the same output.

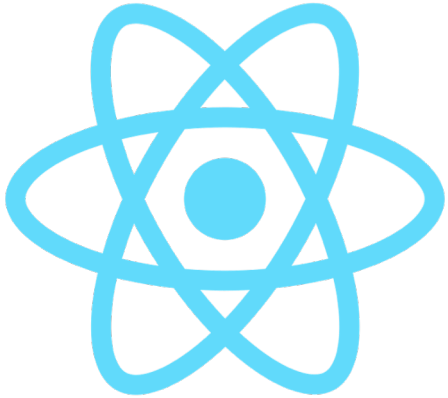
This is how React uses the **reactive pattern**.

What is React?

React is JavaScript

ES6 features

- Destructuring
- Import
- Arrow functions



ES6

Destructuring

ECMAScript5:

```
const title = this.props.title;
const description = this.props.description;
```

ECMAScript6:

```
const {
  title,
  description,
} = this.props;
```

ES6

Import

ECMAScript5:

```
var React = require("react");
console.log(React.PureComponent);
```

ECMAScript6:

```
import { PureComponent } from 'react';
console.log(PureComponent);
```

ES6

Arrow functions

ECMAScript5:

```
const odds = evens.map(function (number) {
  return number + 1;
});
```

ECMAScript6:

```
const odds = evens.map(number => number + 1);
```

Callback heaven

```
<!DOCTYPE html>
<html...>
<body>
    <a href="http://www.google.com/">Click here</a>

    <script>
        document.querySelector('a').addEventListener('click', function(event) {
            event.preventDefault();

            this.style.backgroundColor = this.style.backgroundColor
                    ? null
                    : '#124F86';

            setTimeout(function() {
                const link = this.attributes['href'].value;
                alert('You clicked the link: ' + link);
            }, 16);
        });
    </script>
</body>
```

What is JSX?

JSX is a syntax extension to **JavaScript**.

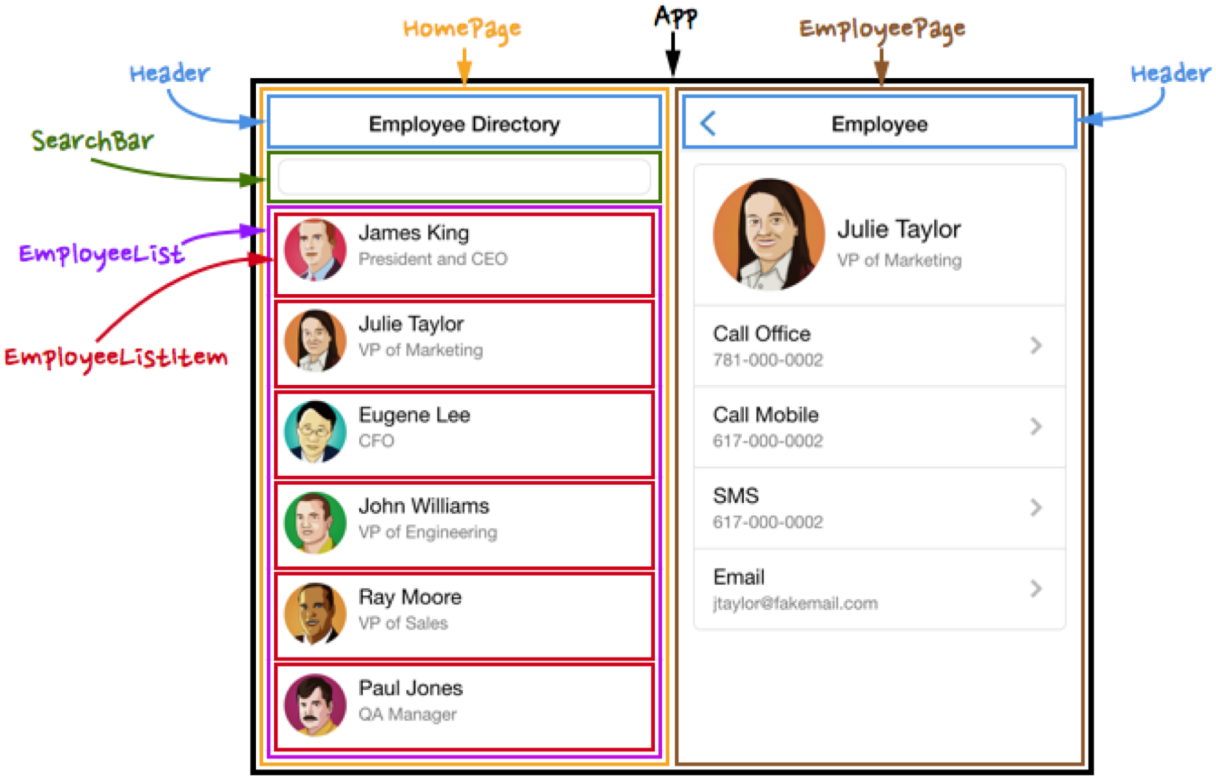
```
const name = 'Obi-Wan Kenobi';

// JSX syntax
const element = (
    <div className="greetings">
        <p>Hello {name}</p>
    </div>
);
```

What is JSX?

JSX is a syntax extension to **JavaScript**.

```
const name = 'Obi-Wan Kenobi';

// JSX syntax
const element = (
    <div className="greetings">
        <p>Hello {name}</p>
    </div>
);
```

What is a React component?



What is a React component?

```
import React, { Component } from 'react';

class Employee extends Component {
    render() {
        return (<div className="employee">
            <h1>Han Solo</h1>
            <p>Smuggler</p>
        </div>)
    }
}
```

Props

```
class Employee extends React.Component {
    render() {
        const {
            name,
            jobTitle,
        } = this.props;

        return (<div className="employee">
            <h1>{name}</h1>
            <p>{jobTitle}</p>
        </div>)
    }
}
```

Rendering a component

```
import React from 'react';
import ReactDOM from 'react-dom';

ReactDom.render(
    <Employee
        name="Han Solo"
        jobTitle="Smuggler"
    />,
    document.getElementById('root')
);
```

PropTypes

```
import PropTypes from 'prop-types';
class Employee extends React.Component {
        static propTypes = {
                name: PropTypes.string.isRequired,
                jobTitle: PropTypes.string,
        }
        render() {
                const {
                        name,
                        jobTitle,
                } = this.props;

                return (<div className="employee">
                        <h1>{name}</h1>
                        <p>{jobTitle}</p>
                </div>)
        }
}
```

PropTypes

```
PropTypes.string

PropTypes.number

PropTypes.bool

PropTypes.func

PropTypes.node

PropTypes.oneOf(['Wookie', 'Ewok'])

PropTypes.arrayOf(PropTypes.number)

PropTypes.shapeOf({
      name: PropTypes.string,
})
```

State

```
class Employee extends React.Component {
    state = { isExpanded: false };
    render() {
        return (<div className="employee">
            <h1>{name}</h1>
            <button onClick={this.onToggle}>
                View more
            </button>
            { this.state.isExpanded
                  && <p>{jobTitle}</p>
            }
        </div>)
    }
}
```

setState()

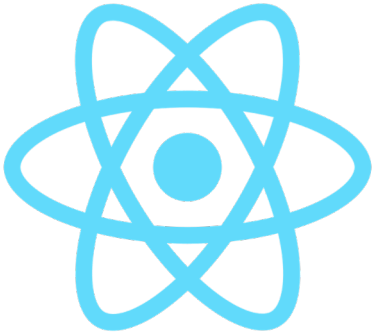
```
class Employee extends React.Component {
        state = { isExpanded: false };
        onExpand = () => {
                this.setState({ isExpanded: true })
        };
        render() {
                return (<div className="employee">
                        <h1>{name}</h1>
                        <button onClick={this.onExpand}>
                                View more
                        </button>
                        { this.state.isExpanded
                                && <p>{jobTitle}</p>
                        }
                </div>)
        }
}
```

setState()

with previous state

```
class Employee extends React.Component {
        state = { isExpanded: false };
        onToggle = () => {
                this.setState((prevState) => {
                        isExpanded: !prevState.isExpanded,
                })
        };
        render() {
                return (<div className="employee">
                        <h1>{name}</h1>
                        <button onClick={this.onToggle}>
                                View more
                        </button>
                        { this.state.isExpanded
                              && <p>{jobTitle}</p>
                        }
                </div>)
        }
}
```

Virtual DOM



- Manipulating the DOM is slow
- Virtual DOM = lightweight copy of the DOM
- Compare virtual DOM before and after updates
- Only the DOM-objects that changed will be updated

→ **Increased performance**

React lifecycle methods

Initialization:

~~**componentWillMount()**~~

render()

componentDidMount()

componentDidMount()

```
class Employee extends React.Component {

      componentDidMount() {

          ga('send',
             'event',
             'mount',
             'employee',
             this.props.name);

      }

      render() {

             return (<div className="employee">

                    <h1>{name}</h1>
                    <p>{jobTitle}</p>

             </div>)

      }

}
```

React lifecycle methods

Props changes:

~~componentWillReceiveProps(nextProps)~~

shouldComponentUpdate(nextProps, nextState)

~~componentWillUpdate(nextProps, nextState)~~

render()

componentDidUpdate(prevProps, prevState)

Refs

Store a reference to a DOM element

```
class CustomForm extends React.Component {
        handleRef = (inputElement) => {
                this.inputElement = inputElement;
        }

        focusOnInput = () => {
                this.inputElement.focus();
        }

        render() {
                return (<div className="custom-form">
                        <input
                                type="text"
                                ref={this.handleRef}
                        />
                        <button onClick={this.focusOnInput}>
                                Focus
                        </button>
                </div>)
        }
}
```

React.Component

```
import React, { Component } from 'react';

class Employee extends Component {
    render() {
        return (<div className="employee">
            <h1>Han Solo</h1>
            <p>Smuggler</p>
        </div>)
    }
}
```

Pure function component

```
import React from 'react’;

const Employee = ({ name, jobTitle }) => (
    <div className="employee">
        <h1>{name}</h1>
        <p>{jobTitle}</p>
    </div>
)
```

React.PureComponent

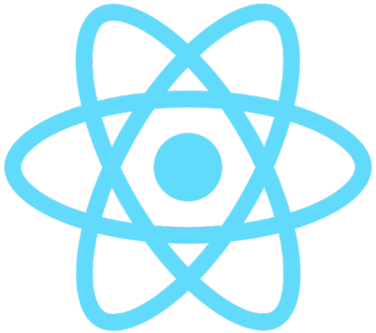
```
import React, { PureComponent } from 'react';

class Employee extends PureComponent {
    render() {
        return (<div className="employee">
            <h1>Han Solo</h1>
            <p>Smuggler</p>
        </div>)
    }
}
```

Break

```
setTimeout(comeBack, 15*60*1e3)
```

Theory summary

- React is a view library
- JSX
- Components
- Props vs. State



Getting started

- Install create-react-app

```
npx create-react-app yourFolderName
```

- Download React Developer Tools
- Navigate to your new folder
- Run your React app

```
npm run start
```

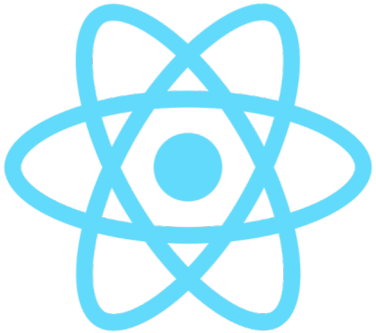
Create-React-App

Your environment will have everything you need to build a modern single-page React app:

- **React**, **JSX** and **ES6** support
- A **live development server** that warns about common mistakes
- A **build script** to bundle JS, CSS, and images for production, with hashes and sourcemaps
- An offline-first **service worker** and a web app manifest, meeting all the Progressive Web App criteria.

What you'll make

- Demo
- React Developer Tools
  - Components
  - Props & State
  - Show updates



Workshop

- Sign in to Github and fork the project https://github.com/uvdata/jarjar-newsfeed

```
git clone git@github.com:USERNAME/jarjar-newsfeed.git
```

- Install the npm packages

```
npm install
```

- Run the app

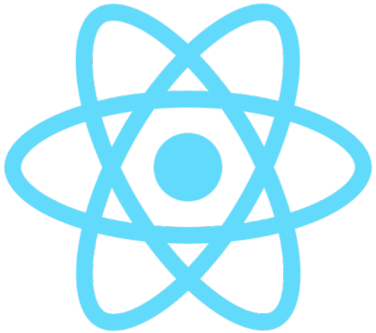
```
npm run start
```

Happy coding!

Develop your components with reusability in mind

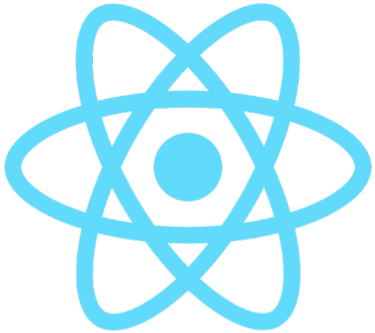
Follow up

- Show and tell?
- Questions?



Curious?

- React meetup
  http://bit.do/aalborg-react
- React native
- Storybook
- Webpack & Babel
- MobX, Redux
- React Router
- Server Side Rendering



DO’s and DON’Ts

- DON’T: use arrow functions or bind in render()
- DO: Write many small components
- DON’T: Use redux/mobx/.. because it’s cool
- DO: Use redux/mobx/.. when you really need it
- DON’T: Dig into the DOM in lifecycle methods that are called on both server and client
- DO: Code Happy!

Christian Lillelund
Head of Development at UVdata
cll@kmd.dk

Contact