React II

Nested Components

Agenda

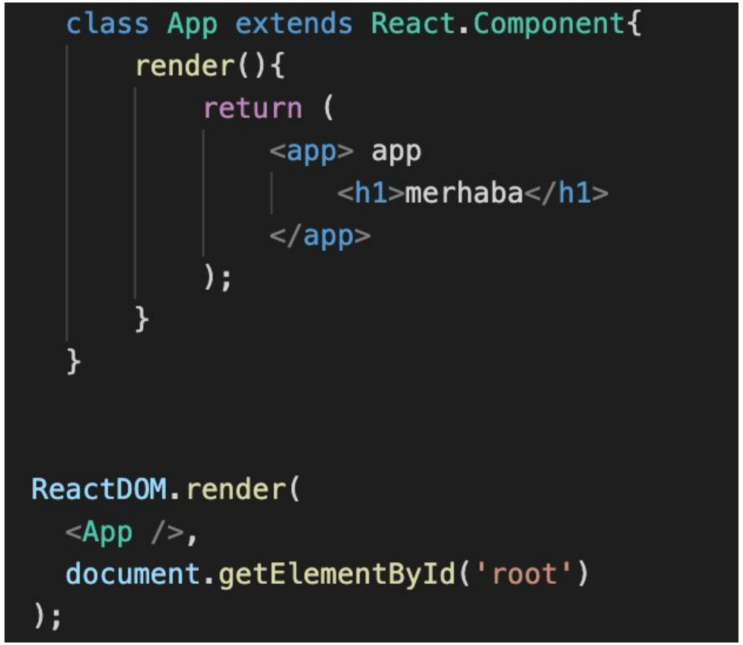
- Functions vs Classes as components
- Nested Components
- Props - pass data into component
- Conditions
- Lists

Learning Objectives

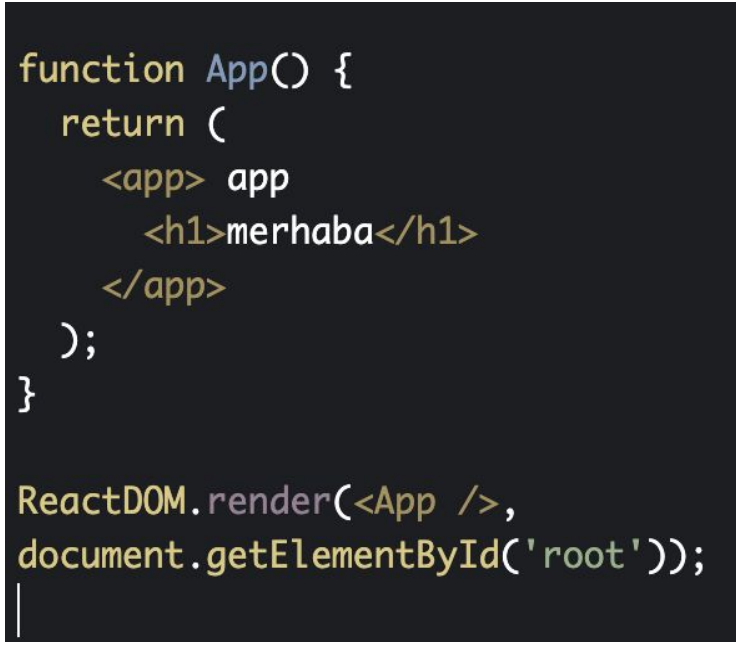
- You know two approaches of react
- You can simply nest components
- You know how data pass through from parent to child component
- You know how to use conditional in JSX
- You know how to use lists in react components

Functions as components

- New! In React you can use functions as components (via JSX)
- It is dead simple to define a component
- It does not have an internal state (state is global referenced)
- Classes can introduce some problems with “this”.
- Classes can give you more performance!
- Facebook officially recommends using functional components wherever possible



```
class App extends React.Component{
    render(){
        return (
            <app> app
                <h1>merhaba</h1>
            </app>
        );
    }
}


ReactDOM.render(
  <App />,
  document.getElementById('root')
);
```



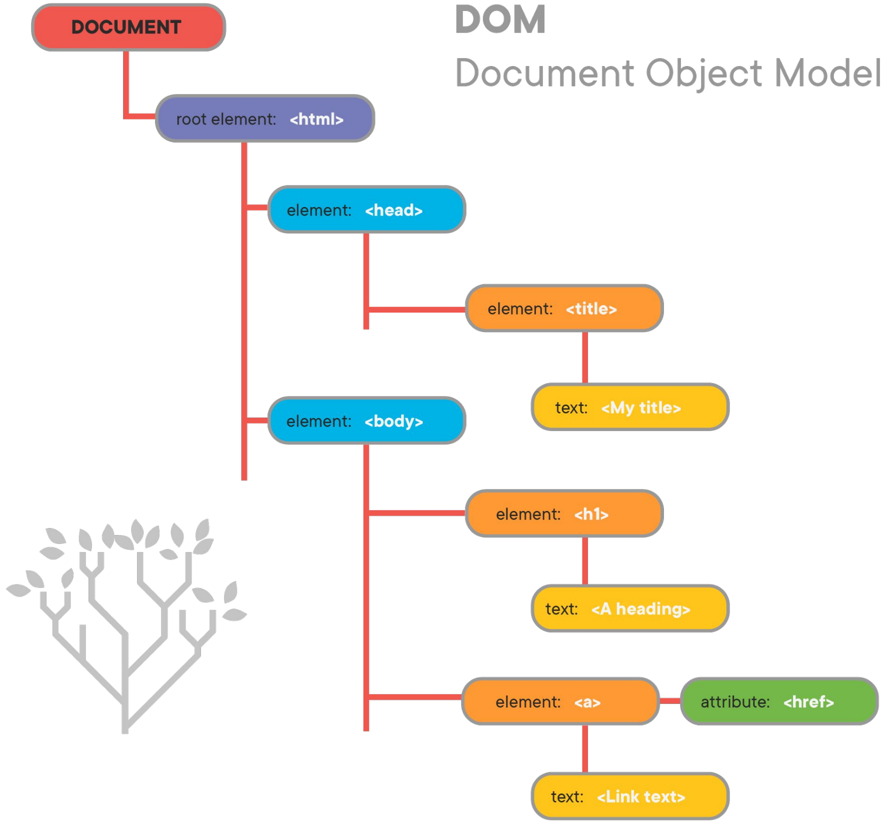
```
function App() {
  return (
    <app> app
      <h1>merhaba</h1>
    </app>
  );
}

ReactDOM.render(<App />,
document.getElementById('root'));
```

Nested Components

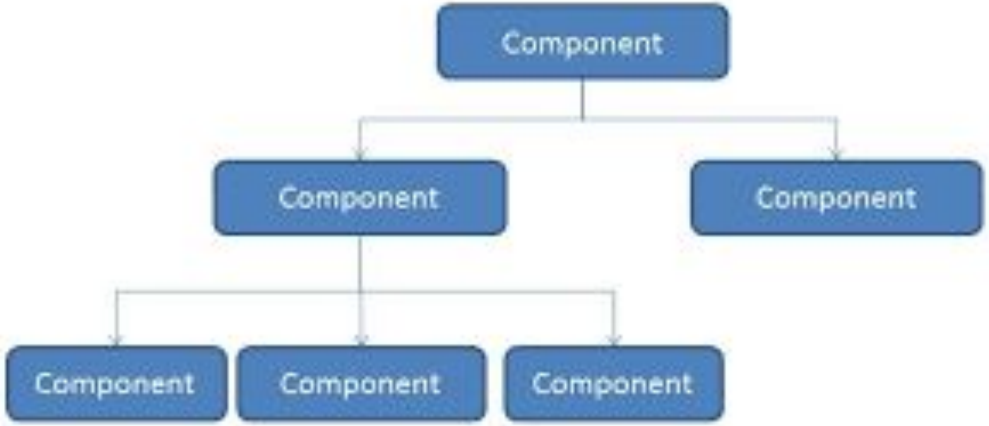
- React is a UI library and builds UI by components in form of a tree like DOM does.
- In DOM, every element has its own children elements (tags, etc..)
- So, the components in React!
- There is no possibility to have more than one root elements at the same time

DOM

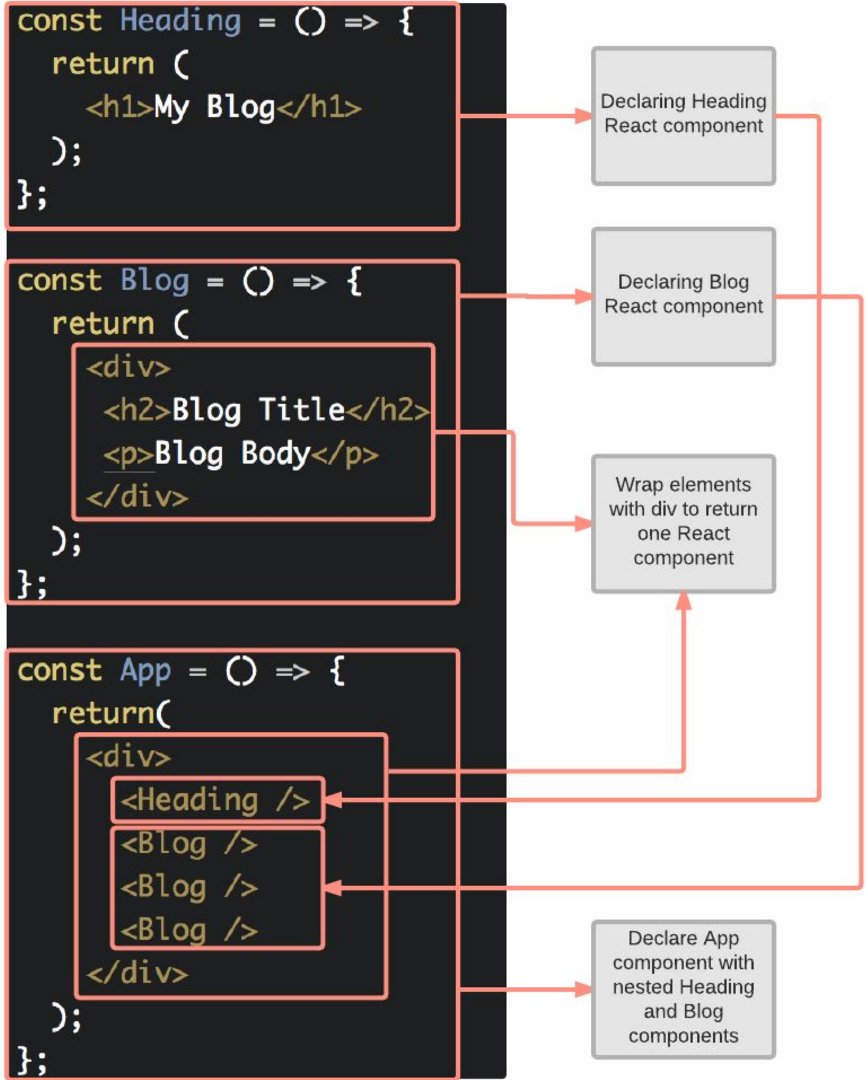
Document Object Model



Source:
https%3A%2F%2Flearn.co%2Flessons%2Ffewpjs-the-dom-tree&psig=AOvVaw1GuYAd6EVIvusVAXWzqKbO&ust=1591512158566000&source=images&cd=vfe&ved=0CA0QjhxqFwoTCND25f3K7OkCFQAAAAAdAAAAABAU



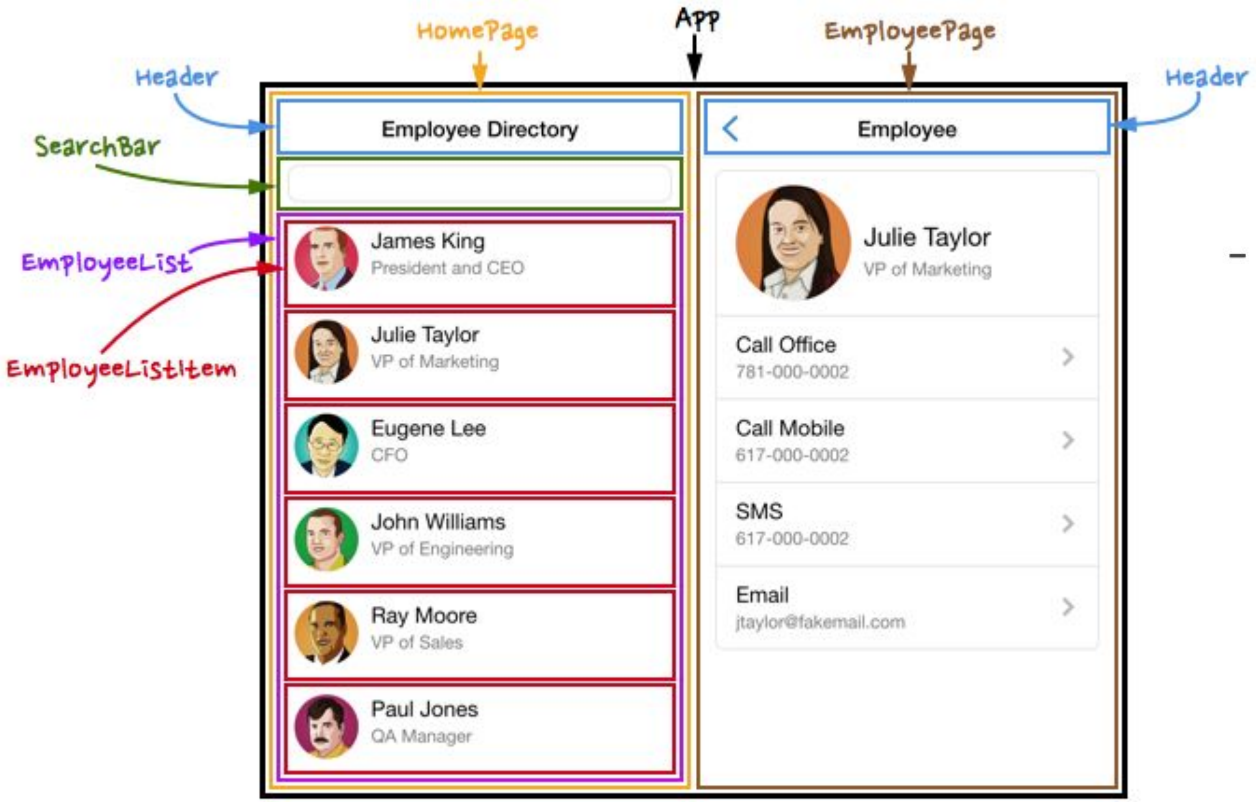
React Virtual DOM



```
const Heading = () => {
  return (
    <h1>My Blog</h1>
  );
};

const Blog = () => {
  return (
    <div>
     <h2>Blog Title</h2>
     <p>Blog Body</p>
    </div>
  );
};

const App = () => {
  return(
    <div>
      <Heading />
      <Blog />
      <Blog />
      <Blog />
    </div>
  );
};
```

Declaring Heading React component
Declaring Blog React component
Wrap elements with div to return one React component
Declare App component with nested Heading and Blog components

Example of nesting



- App
  - HomePage
    - Header
    - SearchBar
    - EmployeeList
      - EmployeeListItem
  - EmployeePage
    - Header
    - EmployeeDetails

hi:coders - Web Shop

Everything what you need

Teddy bär A

CHF 25.-

Teddy bär B

CHF 45.-

Teddy bär C

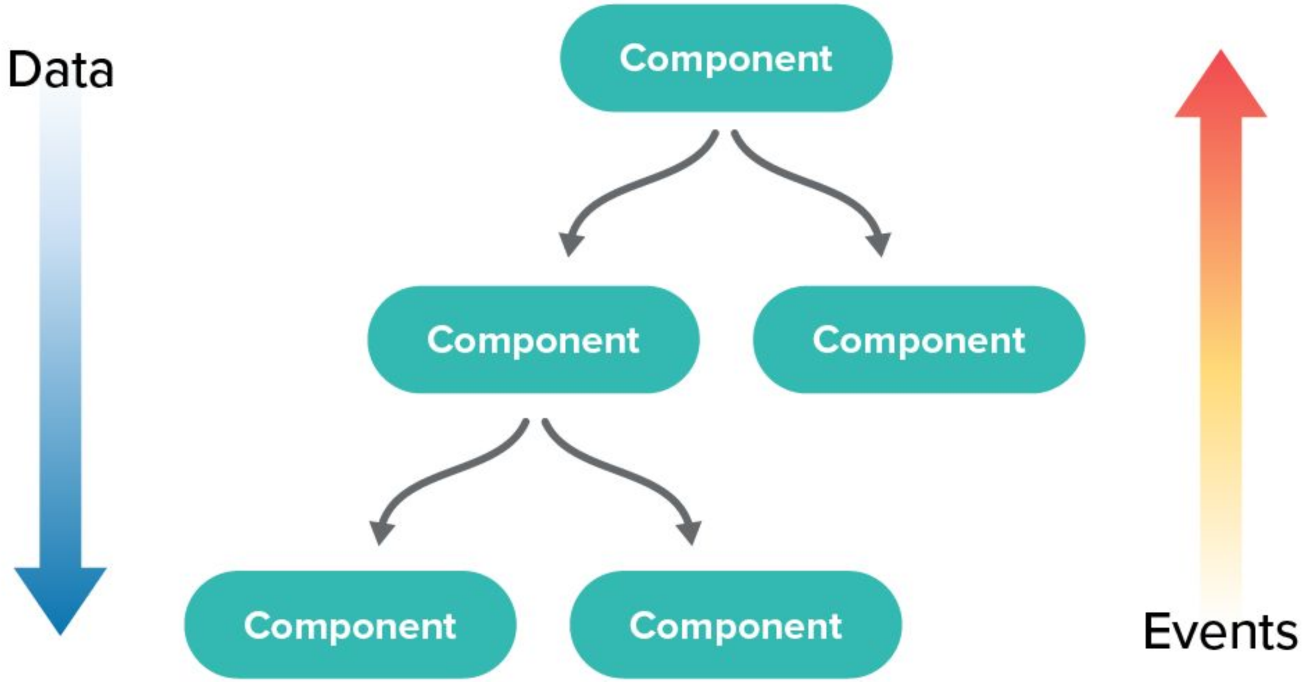
CHF 65.-

Properties – Conditionals – Lists

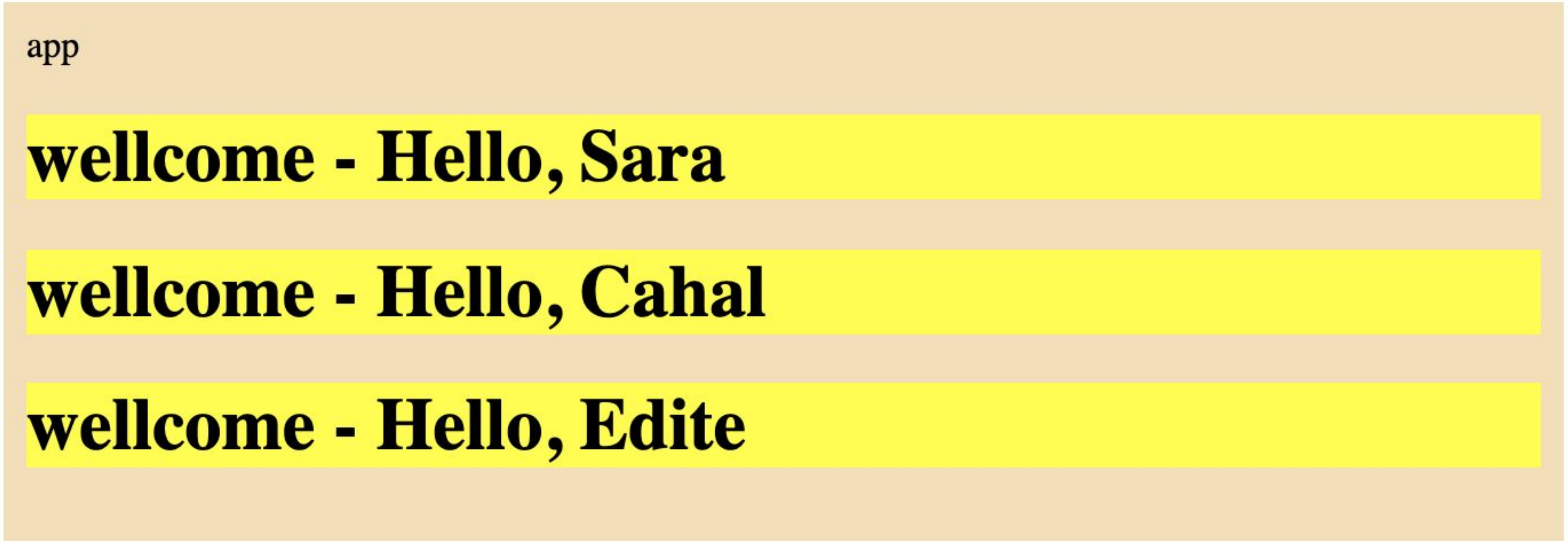
props

How to pass a value?

- Properties on component are passed into internal mechanism.
- They look like html attributes but behave like the function arguments



Direction of passing data

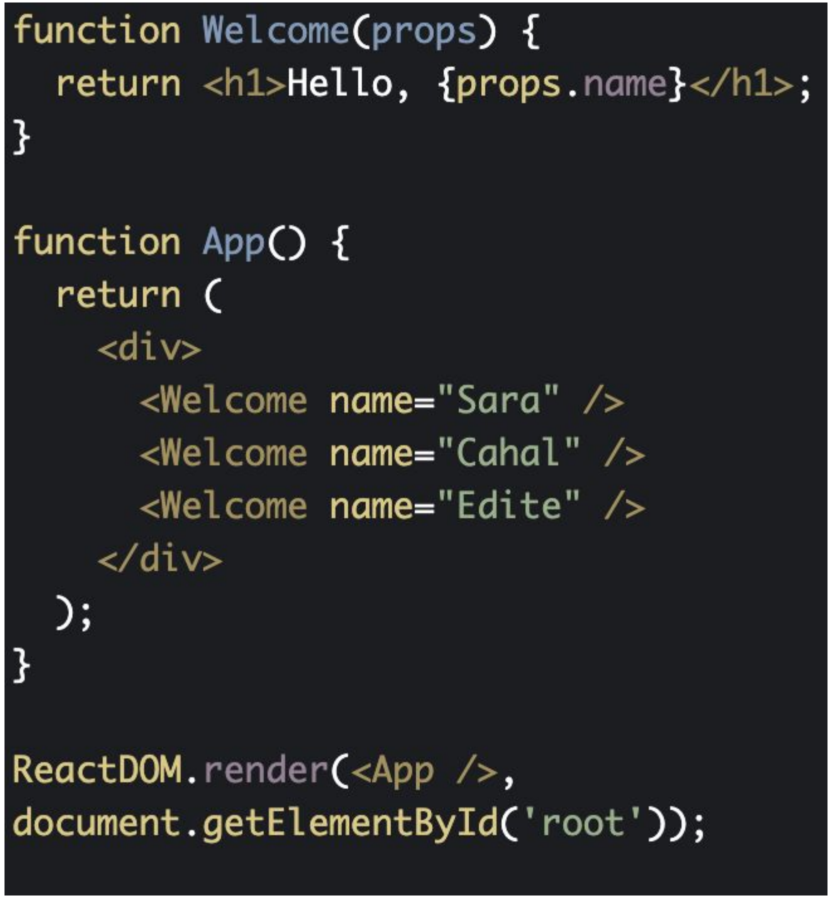

app

wellcome - Hello, Sara

wellcome - Hello, Cahal

wellcome - Hello, Edite

app > welcome



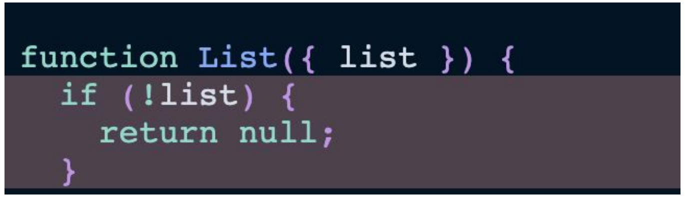
```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}

ReactDOM.render(<App />,
document.getElementById('root'));
```

Solution

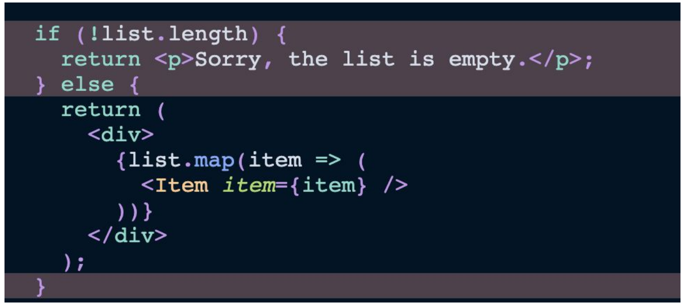
conditions

How to show another html based on a values?

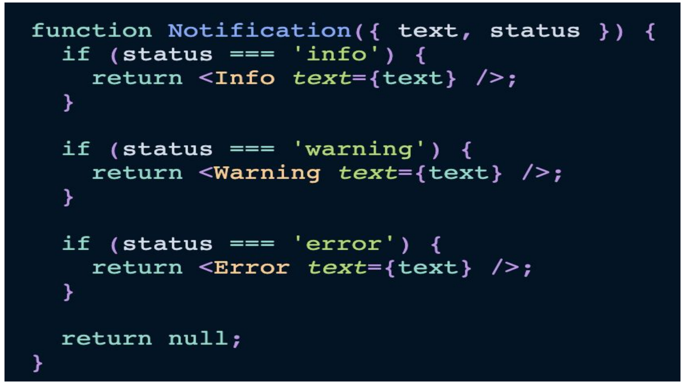
- You need condition logic in a react based component
- There are ways to achieve it
  - Use conditions outside of JSX
  - Use conditions inside of JSX



```
function List({ list }) {
  if (!list) {
    return null;
  }
```

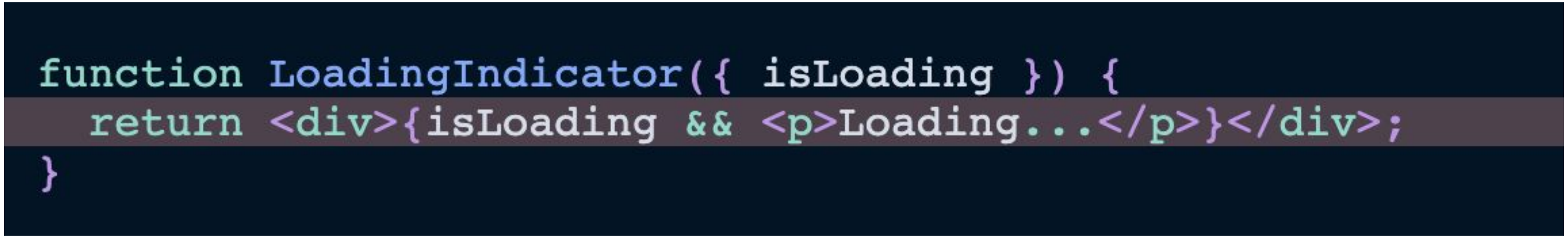


```
if (!list.length) {
  return <p>Sorry, the list is empty.</p>;
} else {
  return (
    <div>
      {list.map(item => (
        <Item item={item} />
      ))}
    </div>
  );
}
```

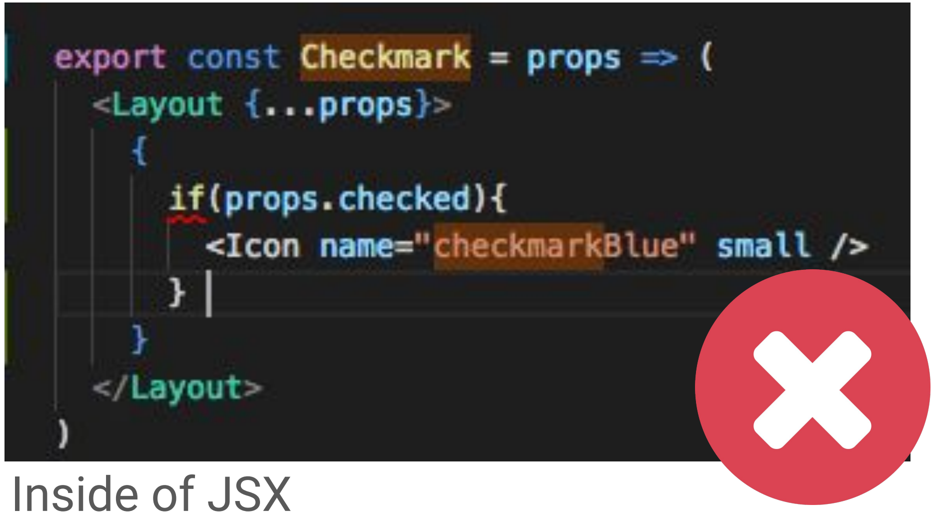


```
function Notification({ text, status }) {
  if (status === 'info') {
    return <Info text={text} />;
  }

  if (status === 'warning') {
    return <Warning text={text} />;
  }

  if (status === 'error') {
    return <Error text={text} />;
  }

  return null;
}
```

Outside of jsx

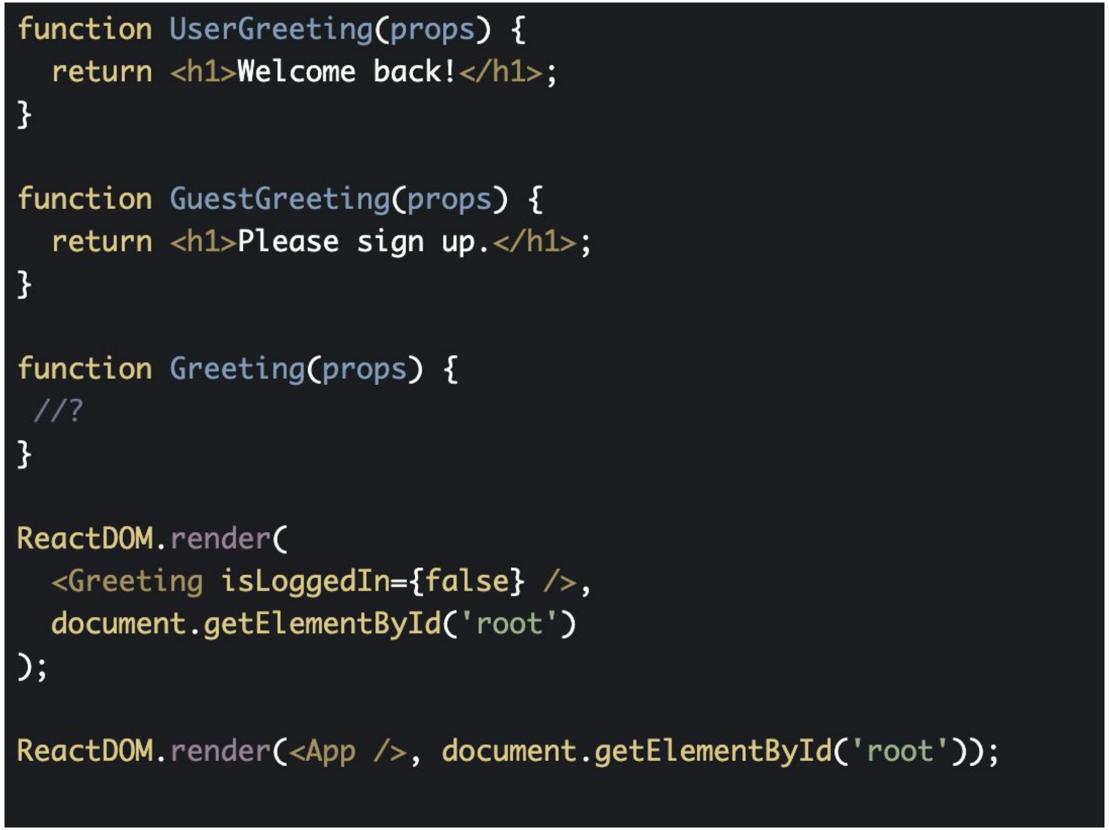


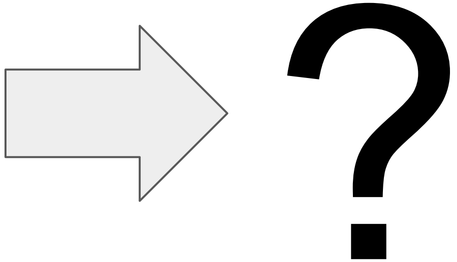
```
function LoadingIndicator({ isLoading }) {
  return <div>{isLoading && <p>Loading...</p>}</div>;
}
```



```
export const Checkmark = props => (
  <Layout {...props}>
    {
      if(props.checked){
        <Icon name="checkmarkBlue" small />
      }
    }
  </Layout>
)
```

Inside of JSX


```
function UserGreeting(props) {
  return <h1>Welcome back!</h1>;
}

function GuestGreeting(props) {
  return <h1>Please sign up.</h1>;
}

function Greeting(props) {
 //?
}

ReactDOM.render(
  <Greeting isLoggedIn={false} />,
  document.getElementById('root')
);

ReactDOM.render(<App />, document.getElementById('root'));
```

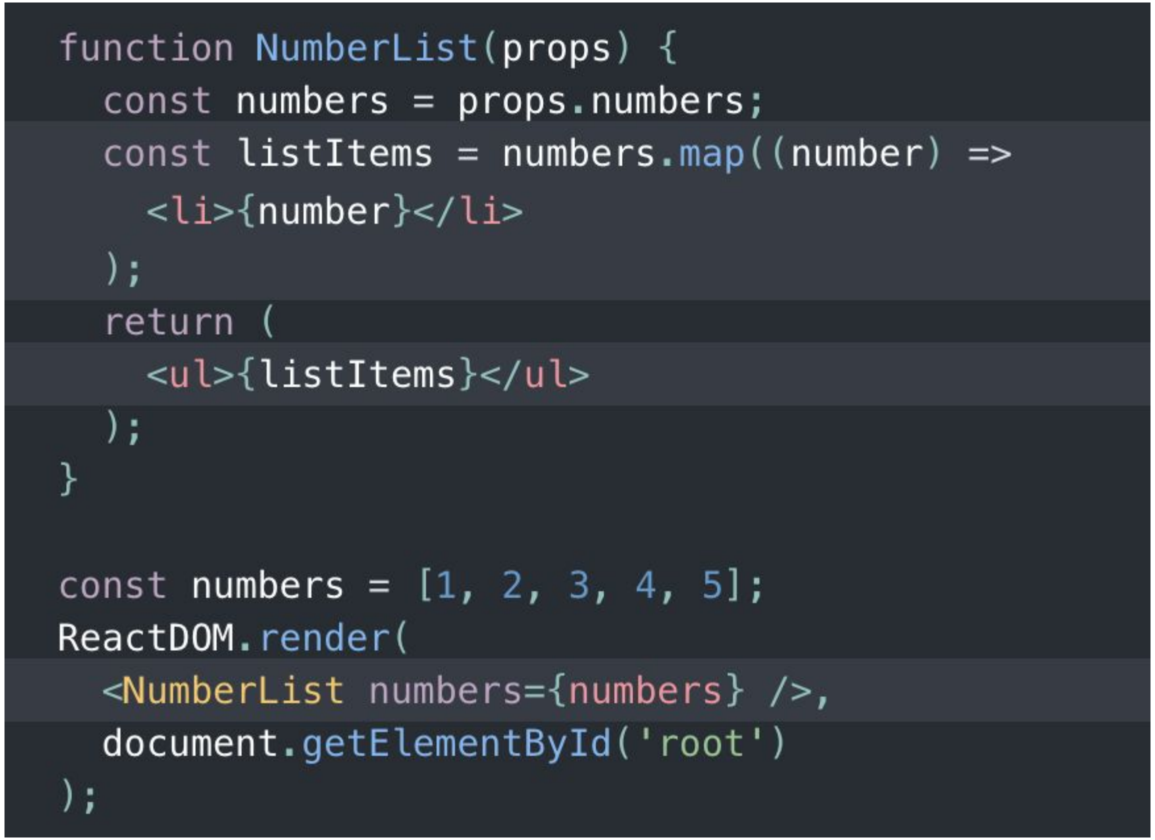


Question

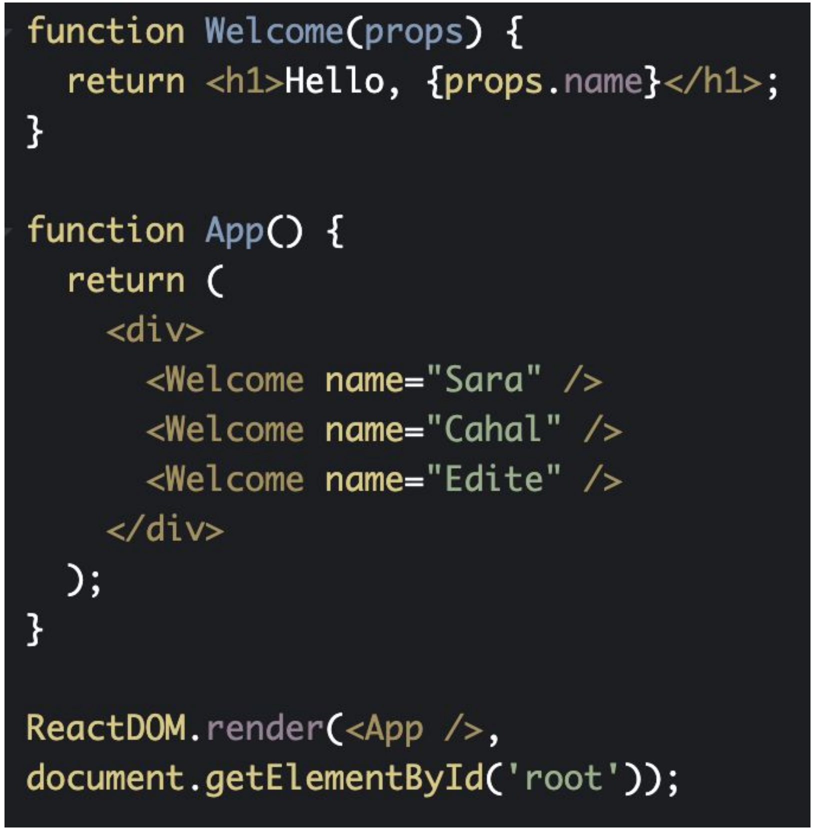
lists

How to work with lists?

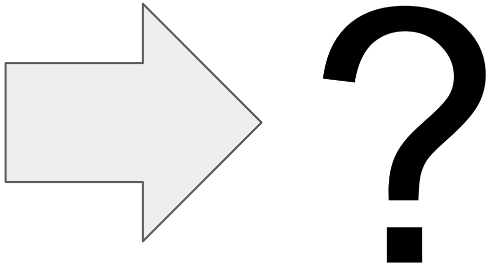
- Rendering lists is generally what you are doing in react apps.
- Use “map” function of array whereas possible
- Uniqueness via “key” property



```
function NumberList(props) {
  const numbers = props.numbers;
  const listItems = numbers.map((number) =>
    <li>{number}</li>
  );
  return (
    <ul>{listItems}</ul>
  );
}

const numbers = [1, 2, 3, 4, 5];
ReactDOM.render(
  <NumberList numbers={numbers} />,
  document.getElementById('root')
);
```



```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}

ReactDOM.render(<App />,
document.getElementById('root'));
```



Write it by using list

Questions