React 101

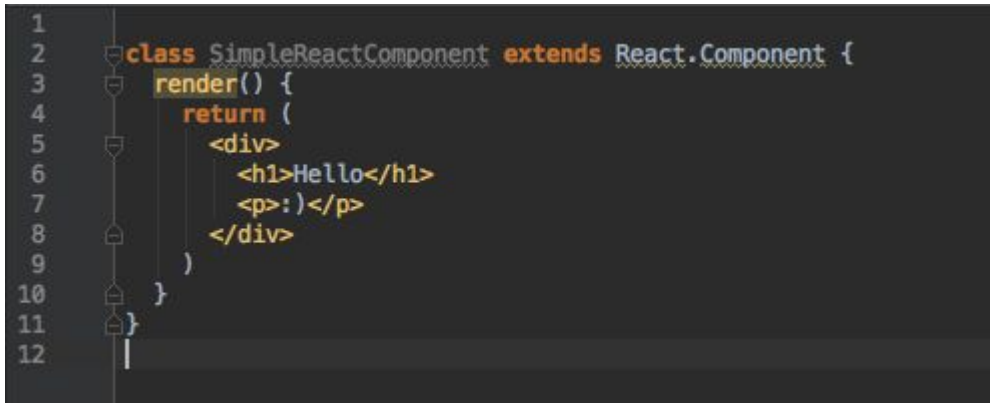
Reminders

- Components
- State

Components

You build functional user interfaces in react using components.

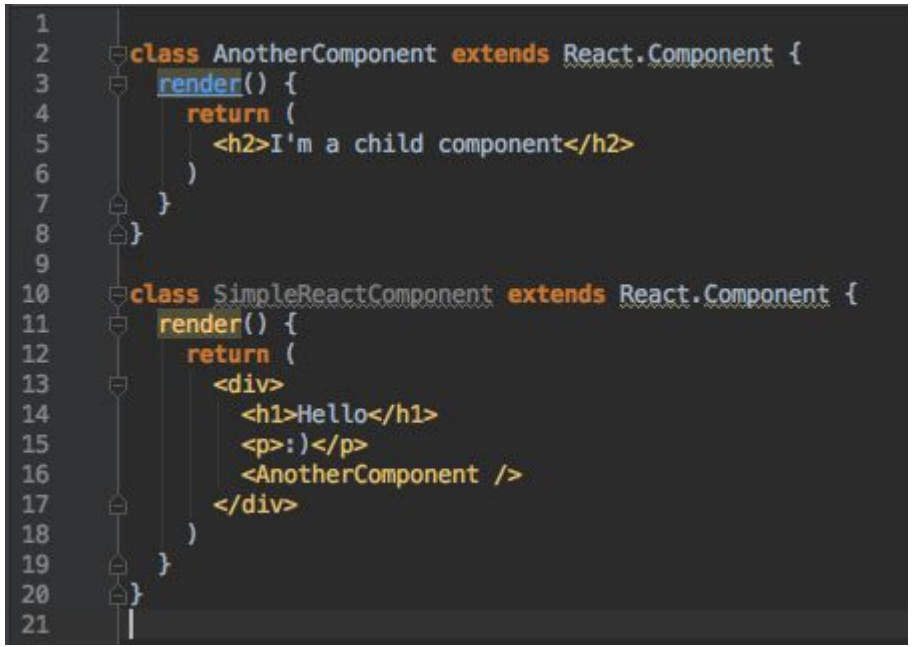
A basic component looks like:



```
class SimpleReactComponent extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello</h1>
        <p>:)</p>
      </div>
    )
  }
}
```

To render html, the component must have a render method.

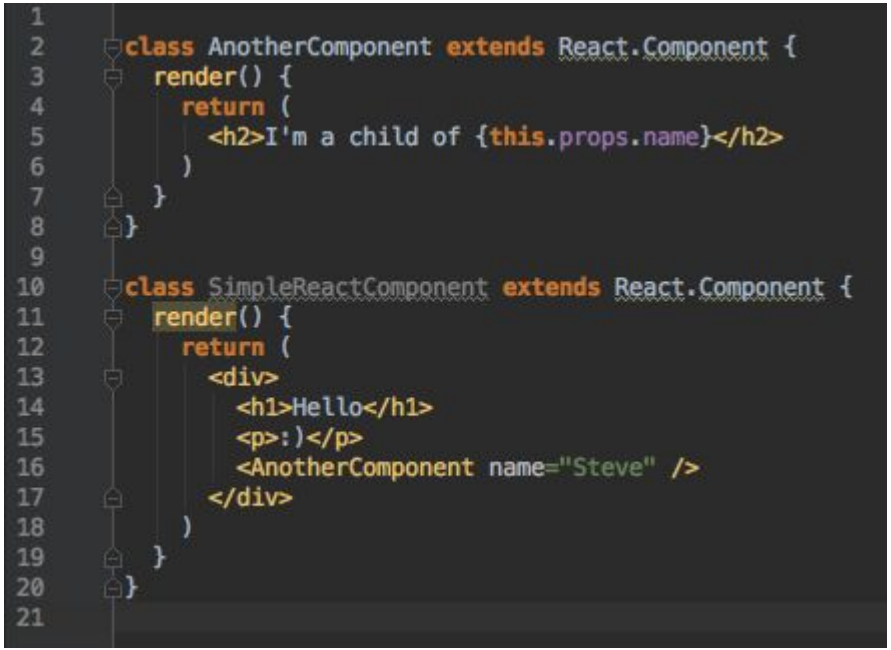
Components can render other components



```
class AnotherComponent extends React.Component {
  render() {
    return (
      <h2>I'm a child component</h2>
    )
  }
}

class SimpleReactComponent extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello</h1>
        <p>:)</p>
        <AnotherComponent />
      </div>
    )
  }
}
```

Components can be passed properties

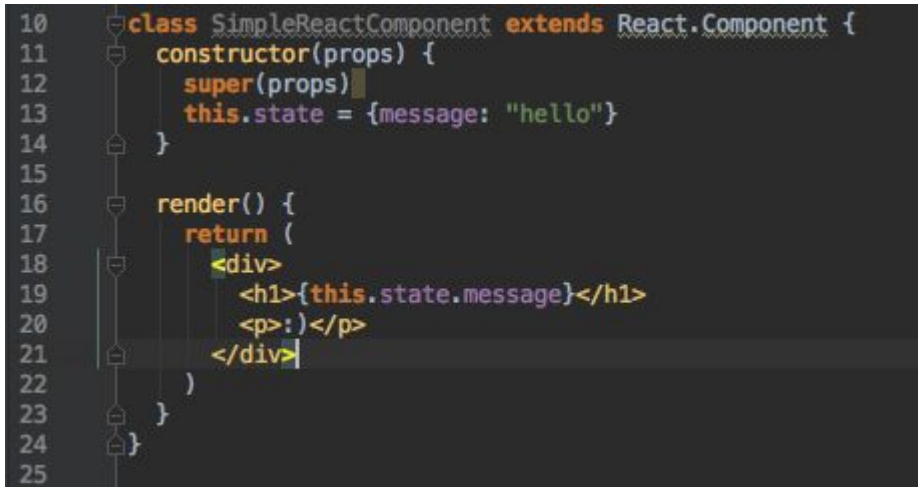


```
class AnotherComponent extends React.Component {
  render() {
    return (
      <h2>I'm a child of {this.props.name}</h2>
    )
  }
}

class SimpleReactComponent extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello</h1>
        <p>:)</p>
        <AnotherComponent name="Steve" />
      </div>
    )
  }
}
```

State

State is just an instance variable of the component. Changing the value of that variable will cause the component to re-render.



Updating state

```
4   class SimpleReactComponent extends Component {
5     constructor(props) {
6       super(props)
7       this.state = {numberOfClicks: 0}
8     }
9
10    incrementClicks = () => {
11      this.setState({numberOfClicks: this.state.numberOfClicks + 1})
12    }
13
14    render() {
15      return (
16        <div className="App">
17            <h2>React workshop 2</h2>
18            <p>{this.state.numberOfClicks}</p>
19            <button onClick={this.incrementClicks}>Click</button>
20        </div>
21      );
22    }
23  }
24
25  export {SimpleReactComponent};
26
```

You can update state in methods.
You can bind methods to html events (onClick, onChange)

Reminder: **Because of scoping issues, methods used on html events that update state must be arrow function instead of regular.**

myFunction = () => {

Not

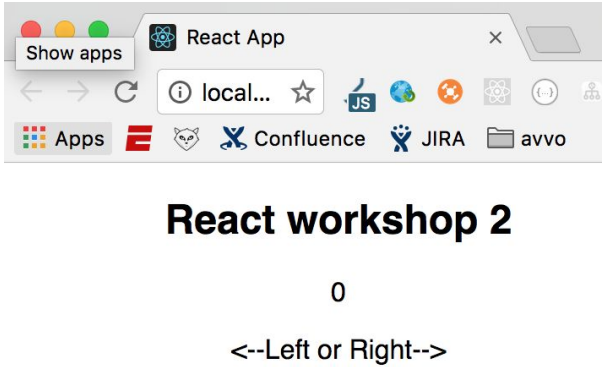
myFunction() {

Fun new stuff

- Updating a parent’s state in a child
- Storing html element in variable

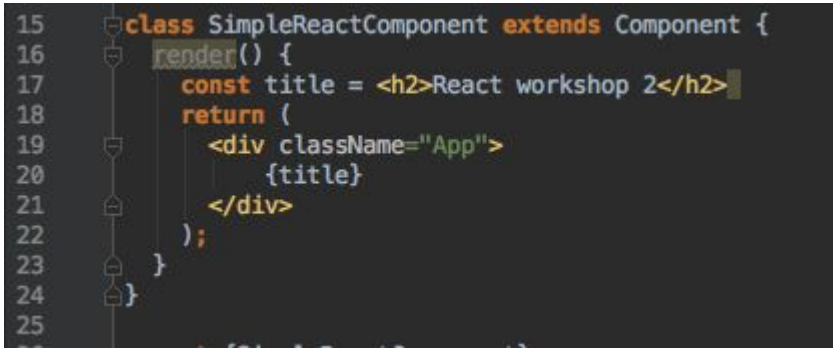
Updating parent’s state in child

Methods that update state can be passed to children and called




```
3   class ChildComponentArrows extends Component {
4     render() {
5       return (
6         <div>
7           <label onClick={this.props.onClickLeft}>&lt;--</label>
8           Left or Right
9           <label onClick={this.props.onClickRight}>--&gt;</label>
10        </div>)
11    }
12  }
13
14
15  class SimpleReactComponent extends Component {
16    constructor(props) {
17      super(props)
18      this.state = {numberOfClicks: 0}
19    }
20
21    decrement = () => {
22      this.setState({numberOfClicks: this.state.numberOfClicks - 1})
23    }
24
25    increment = () => {
26      this.setState({numberOfClicks: this.state.numberOfClicks + 1})
27    }
28
29    render() {
30      return (
31        <div className="App">
32            <h2>React workshop 2</h2>
33            <p>{this.state.numberOfClicks}</p>
34            <ChildComponentArrows onClickLeft={this.decrement} onClickRight={this.increment} />
35        </div>
36      );
37    }
38  }
39
```

Storing html element in variable



Storing html element in variable cont

```
15  class SimpleReactComponent extends Component {
16    render() {
17      const items = ["item1", "item2", "item3"]
18      const itemList = items.map(item => {return <li>{item}</li>})
19      return (
20        <div className="App">
21            <ul>
22              {itemList}
23            </ul>
24        </div>
25      );
26    }
27  }
28
29  export {SimpleReactComponent};
30
```

Or

```
15  class SimpleReactComponent extends Component {
16    render() {
17      const items = ["item1", "item2", "item3"]
18      return (
19        <div className="App">
20            <ul>
21              {items.map(item => {return <li>{item}</li>})}
22            </ul>
23        </div>
24      );
25    }
26  }
27
28  export {SimpleReactComponent};
29
```

Html outside of return

Logic inside of return

Your turn - Type ahead

- Grab the react workshop https://github.com/tswayne/react-workshop
- Update the app to
  - Have an input
  - Under the input - Show a list of game titles that match the text in the input
- If you finish - Make the input and list each a separate component
- If you finish that - Use video-game-objs instead of titles

Hint

```
11    myOnChange = (e) => {
12      console.log(e.target.value + 'is the value of the text in the input')
13    }
14
15
16    render() {
17      return (
18        <div className="App">
19            <input onChange={myOnChange} />
```