AW Academy

React (Part 2)

Class Components and Event Handling

Learning Outcomes

- Understands class components
  - State
  - Lifecycle methods
- Implements event handling
  - Using arrow functions in class components

Components as Classes

- https://reactjs.org/docs/components-and-props.html#function-and-class-components

- We can do the same things with both
- Some projects use functions, some classes and others a mix
- You shoud be prepared to work with both

React Class Components

- Extend React.Component
- Props are stored in this.props
- Return Elements to display from render Method

Converting to Class Component

- Convert the Article component to a class

```
class Article extends React.Component {
  /* TODO... */
}
```

- The rest of the code stays unchanged

React Event Handlers

- Pass a function as a prop or HTML event attribute to have it called on some event.
  - Reminder: Props can be any type, including Functions

https://reactjs.org/docs/handling-events.html

Handling a button click

- Copy the code to the right into your App.js
- Insert the component ClickCounter somewhere in your App
- Click the button and observer in your browser's console what happens

Why does the counter not increase?

```
class ClickCounter extends React.Component {
  render() {

    let counter = 0;

    const clickHandler = () => {
      console.log("Button Clicked");
      counter++;
    }

    console.log("Rendering...")
    return <div>
      <button onClick={clickHandler}>
        Click me!
      </button>
      You have clicked {counter} times.
    </div>
  }
}
```

Fixing the Counter - Attempt 1

- Create a class field **`counter`** and replace all uses of the local variable **`counter`** with **`this.counter`**
- Log the value of **`this.counter`** inside of **`clickHandler`**
- Does it work now?

When do Components re-render?

- When the Parent Component re-renders (possibly changing the props)
- When the **State** changes

State

- An object which can be filled with arbitrary values (similar to props)
- Set and updated by the component itself (props only change from outside)

Initially set in the class constructor

State

Initial State is set in the class constructor

```
class ClickCounter extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      counter: 0,
    };
  }
  // ...
}
```

(When defining our own constructor we need to call the **`super`** constructor for `React.Component`)

Reading State

Read State directly from this.state:

```
You have clicked {this.state.counter} times.
```

But do not try to write to **`this.state`** directly!

```
// Will not work
this.state.counter = this.state.counter + 1;
```

Writing State

- Call this.setState with an object containing all the state properties you wish to change:

```
this.setState({
  counter: this.state.counter + 1,
});
```

Fixing the Counter

- Using the Code from the previous slides fix the counter by storing the number of clicks in the state!

Conditional Rendering

Let's go back to the Article Component. Every article has a content (its children).

Implement behavior: To see the content, a user must click on the article's title.

Hints:

- State contains information: "Has the user clicked the title" true or false
- You can put React Elements in variables
  - use curly brackets in JSX Expressions to include
- You can use **`if`** inside the render method (but only before **`return`**)
- **`null`** and **`undefined`** are valid as React Elements - they will display nothing

Component Lifecycle

- React components don't live forever, they may be added or removed (by parent components)
- Phases of component lifecycle:
  - Mounting
  - Updating
  - Unmounting
- A component instance mounts and unmounts exactly once, but can update many times

Common Lifecycle Methods

- Mounting
  - constructor
  - render
  - componentDidMount
- Updating
  - render
  - componentDidUpdate
- Unmounting
  - componentWillUnmount

https://reactjs.org/docs/react-component.html#componentdidmount

https://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/

StopWatch Component

Create a component which displays the number of seconds since it has been mounted

Hints:

- Use setInterval in componentDidMount
- Clean up after yourself: Use clearInterval in componentWillUnmount

Use the StopWatch Component in multiple places, including as a child of Article Components

Learning Outcomes Review

- Understands class components
  - State
  - Lifecycle methods
- Implements event handling
  - Using arrow functions in class components