WILL WE EVER USE REACT HOOKS?

WHAT ARE REACT HOOKS?

- Feature proposal
- Currently on React v16.7.0-alpha

BUT... WHY?

CLASS COMPONENTS PROBLEMS

- Huge components
- Duplicated logic
- Complex patterns

WHAT IS A HOOK?

- A function that has a state and reacts whenever the state changes.

HOOKS EXAMPLE

- useState(initialState)
  - Initial state doesn’t have to be an object
  - Returns an array containing the current state and a function that updates it
  - Always replace the state value

```jsx
import { useState } from 'react';

function Example() {
    const [count, setCount] = useState(0);

    return (
        <div>
            <p>You clicked {count} times</p>
            <button onClick={() => setCount(count + 1)}>
            Click me
            </button>
        </div>
    );
}
```

HOOKS EXAMPLE

- useEffect(callback)
  - componentDidMount, componentDidUpdate, and componentWillUnmount combined
  - Calls `callback` after each render

```jsx
import { useState, useEffect } from 'react';

function Example() {
    const [count, setCount] = useState(0);

    useEffect(() => {
        document.title = `You clicked ${count} times`;
    });

    return (
        <div>
            <p>You clicked {count} times</p>
            <button onClick={() => setCount(count + 1)}>
            Click me
            </button>
        </div>
    );
}
```

OTHER HOOKS

- useReducer(reducer, initialState)
- useCallback(callback)
- useRef()

CUSTOM HOOKS

```
class App extends React.Component {
    state = {
        nume: 'Pasare',
        prenume: 'Ionut',
        varsta: 22
    }

    handleNume = nume => {
        this.setState({nume});
    }

    handlePrenume = prenume => {
        this.setState({prenume});
    }

    handleVarsta = varsta => {
        this.setState({varsta});
    }

    logValues = () => {
        console.log(nume.value, prenume.value, varsta.value)
    }

    render() {
        let {nume, prenume, varsta} = this.state;

        return (
            <div>
                <input value={nume} onChange={this.handleNume} />
                <input value={prenume} onChange={this.handlePrenume} />
                <input value={varsta} onChange={this.handleVarsta} />

                <button onClick={this.logValues}>VALUES</button>
            </div>
        )
    }
}
```

CUSTOM HOOKS

```
1  import React, { useState, useCallback } from 'react'
2
3  const App = () => {
4      let nume = useTextInput('Pasare'),
5          prenume = useTextInput('Ionut'),
6          varsta = useTextInput(22),
7          logValues = () => console.log(nume.value, prenume.value, varsta.value);
8
9      return (
10         <div>
11             <input {...nume} />
12             <input {...prenume} />
13             <input {...varsta} />
14
15             <button onClick={logValues}>VALUES</button>
16         </div>
17     )
18 }
19
20 const useTextInput = (initialValue = '') => {
21     let [value, setValue] = useState(initialValue);
22
23     return {
24         onChange: useCallback(e => setValue(e.target.value)),
25         value
26     }
27 }
```

PROS

- No breaking changes
- Less code
- Smaller builds
- Code easier to read
- Reusable code
- They’re functions

CONS

- No hooks for `getSnapshotBeforeUpdate` and `componentDidCatch`
- New approach

SO... WILL WE EVER USE THEM?

THANK YOU