Oleksandr Oleksiv

Frontend developer

React Hooks

What are React Hooks?

React Hooks are functions that allow to "power up" your functional components with class-only features (e.g. state, lifecycle hooks, etc.)

Functional	Class-based
Great for presentation	Hold the business logic
Properties as input, JSX as output	Work with props and state
Single purpose focus	Components orchestration

Problems

Wrapper hell

Extending components with new functionality

Lifecycle hooks

Components with Hooks

- Easier to maintain
- Flexible
- Easier to manage lifecycle hooks
- Less code

useState(...)

```
export default class Todo extends Component {
    state = {
        inputValue: "",
        todoList: []
    };

    onChangeHandler = (event) => {
       this.setState({
           inputValue: event.target.value
       })
    };

    onClickHandler = () => {
        this.setState({
            todoList: this.state.todoList.concat(this.state.inputValue),
            inputValue: ""
        })
    };

    render() {
        const todos = this.state.todoList.map((item, index) => <li key={index}
                                                                   className="todo-list__item">{item}</li>);
        return (
            <Fragment>
                <input type="text"
                       className="input-control"
                       placeholder="Todo"
                       value={this.state.inputValue}
                       onChange={this.onChangeHandler}/>
                <button type="button"
                        className="btn btn-default"
                        disabled={!this.state.inputValue.length}
                        onClick={this.onClickHandler}>Add to list</button>
                <ul className="todo-list">
                    {todos}
                </ul>
            </Fragment>
        )
    }
}
```

```jsx
const Todo = () => {
    const [inputValue, setInputValue] = useState('');
    const [todos, setTodos] = useState([]);

    const onChangeHandler = (event) => {
        setInputValue(event.target.value)
    };

    const onClickHandler = () => {
       setTodos(todos.concat(inputValue));
       setInputValue("");
    };

    const todoListItems = todos.map((item, index) => <li key={index} className="todo-list__item">{item}</li>);

    return (
        <Fragment>
            <input type="text"
                   placeholder="Type here"
                   className="input-control"
                   value={inputValue}
                   onChange={onChangeHandler}/>
            <button type="button"
                    className="btn btn-default"
                    onClick={onClickHandler}>Add</button>
            <ul>
                {todoListItems}
            </ul>
        </Fragment>
    );
};
```

```
const Todo = () => {
    const [todoState, setTodoState] = useState({
        inputValue: '',
        todos: []
    });

    const onChangeHandler = (event) => {
        setTodoState({
            inputValue: event.target.value,
            todos: todoState.todos
        })
    };

    const onClickHandler = () => {
        setTodoState({
            todos: todoState.todos.concat(todoState.inputValue),
            inputValue: ''
        });
    };

    const todoListItems = todoState.todos.map((item, index) => <li key={index} className="todo-list__item">{item}</li>);

    return (
        <Fragment>
            <input type="text"
                   placeholder="Type here"
                   className="input-control"
                   value={todoState.inputValue}
                   onChange={onChangeHandler}/>
            <button type="button"
                    className="btn btn-default"
                    onClick={onClickHandler}>Add</button>
            <ul>
                {todoListItems}
            </ul>
        </Fragment>
    );
};
```

useEffect(...)

```
export default class Todo extends Component {
    state = {
        inputValue: "",
        todoList: []
    };

    componentDidMount() {
        axios.get('api/v1/todos.json')
            .then(todos => {...})
            .catch(error => console.error(error));
    }

    onChangeHandler = (event) => {
       this.setState({
           inputValue: event.target.value
       })
    };

    onClickHandler = () => {
        axios.post('api/v1/todos.json', {name: this.state.inputValue})
            .then(todos => {...})
            .catch(error => console.error(error));
    };

    componentWillUnmount() {
        //    cleanup stuff goes here
    }

    render() {
        const todos = this.state.todoList.map((item, index) => <li key={index}
                                                                   className="todo-list__item">{item}</li>);
        return (
            <Fragment>
                <input type="text"
                       className="input-control"
                       placeholder="Todo"
                       value={this.state.inputValue}
                       onChange={this.onChangeHandler}/>
                <button type="button"
                        className="btn btn-default"
                        disabled={!this.state.inputValue.length}
                        onClick={this.onClickHandler}>Add to list</button>
                <ul className="todo-list">
                    {todos}
                </ul>
            </Fragment>
        )
    }
}
```

```
const Todo = () => {
    const [inputValue, setInputValue] = useState('');
    const [todos, setTodos] = useState([]);

    useEffect(() => {
        axios.get('api/v1/todos.json')
            .then(todos => {
                setTodos(todos);
            })
            .catch(error => console.error(error));
    }, []);

    const onChangeHandler = (event) => {
        setInputValue(event.target.value)
    };

    const onClickHandler = () => {
        axios.post('api/v1/todos.json', {name: inputValue})
            .then(todos => {
                setTodos(todos);
                setInputValue("");
            })
            .catch(error => console.error(error));
    };

    const todoListItems = todos.map((item, index) => <li key={index} className="todo-list__item">{item}</li>);

    return (
        <Fragment>
            <input type="text"
                   placeholder="Type here"
                   className="input-control"
                   value={inputValue}
                   onChange={onChangeHandler}/>
            <button type="button"
                    className="btn btn-default"
                    onClick={onClickHandler}>Add</button>
            <ul>
                {todoListItems}
            </ul>
        </Fragment>
    );
};
```

```
useEffect(() => {
    window.addEventListener('mousemove', onMouseMoveHandler);
    return () => {
        window.removeEventListener('mousemove', onMouseMoveHandler);
    }
});

const onMouseMoveHandler = event => {
    console.log(event.clientX, event.clientY);
};
```

useContext(...)

```jsx
const App = () => {
  const [view, setView] = useState('todos');
  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const onViewChangeHandler = (viewName) => {
    setView(viewName);
  };

  return (
    <div className="App">
        <AuthContext.Provider value={{status: isLoggedIn, login: setIsLoggedIn}}>
            <Header onLoadTodos={() => onViewChangeHandler('todos')}
                    onLoadAuth={() => onViewChangeHandler('auth')} />
            {view === 'todos' ? <Todo/> : <Auth />}
        </AuthContext.Provider>
    </div>
  );
};
```

```jsx
const Auth = props => {
    const auth = useContext(AuthContext);
    return (
        <div>
            <h1>This is Auth component</h1>
            <button onClick={auth.login}>Login</button>
        </div>
    )
};
```

```jsx
const Header = props => {
    const auth = useContext(AuthContext);
    return (
        <header>
            {auth.status
                ? <button type="button" onClick={props.onLoadTodos}>Todos</button>
                : <button type="button" onClick={props.onLoadAuth}>Authenticate</button>
            }
        </header>
    );
};
```

useMemo(...)

```
33          <Fragment>
34              <input type="text"
35                     placeholder="Type here"
36                     className="input-control"
37                     value={inputValue}
38                     onChange={onChangeHandler}/>
39              <button type="button"
40                      className="btn btn-default"
41                      onClick={onClickHandler}>Add</button>
42              <ul>
43                  {useMemo(() => <TodoList items={todos} />, [todos])}
44              </ul>
45          </Fragment>
```

Rules of Hooks

All hooks should be used at top level

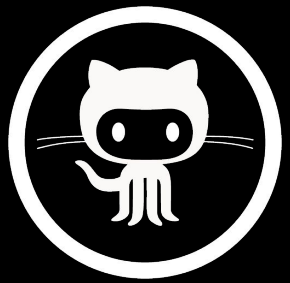
Hook’s can’t be declared conditionally

Use hooks ONLY inside functional components and custom hooks

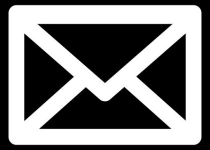
The order matters

Thank You

Contact me



github.com/OleksivO



Oleksandr.Oleksiv
@scalac.io