STRONGBYTES

REACT COMPONENT PATTERNS

As React slowly became the de facto choice for multiple greenfield projects, certain patterns started to emerge in order to write clean, maintainable, better code. These are often viewed as pillars of software quality, presenting huge advantages especially when scalability of the codebase is involved.

In this training we will explore various React component patterns, starting with less complicated examples (stateless functions, conditional rendering, etc.) and moving towards more advanced concepts such as Higher Order Components (HOCs) or functions as child components. The primary goal of this training will be to embed a functional, hooks-based approach towards building UIs using React. Feel free to consult the high-level agenda below.

INTENDED AUDIENCE

This training assumes the participants are already familiar with ES6 concepts (classes, destructuring, default parameters, arrow functions, let/const, etc.), basic JavaScript knowledge and syntax, npm (interacting with the npm ecosystem, package.json), Git (basic commands) and that they have basic React and Redux knowledge (JSX, VDOM, state, props, event management, component lifecycle hooks, actions, reducers, store, mapStateToProps, mapDispatchToProps etc.). Thus, the aforementioned concepts will not be discussed or detailed during the training.

This training targets intermediate developers who would like to upgrade their React knowledge by finding out more about React component patterns, when and how to

apply them and gain architectural knowledge useful in scaling applications based on this technical stack.

OUTLINE

Included below are the main topics covered in this training. The duration is 2 days.

1. Stateless functions
    a. What are stateless functions?
    b. Hooks and performance considerations
2. JSX spread attributes
3. Destructuring arguments
4. Conditional rendering
    a. If
    b. Unless
    c. If / Else block
5. Arrays as children
    a. Usage with functional methods (map, filter, etc.)
    b. Explanation of React keys
6. Controlled vs. Uncontrolled components

    a. What are Controlled components?

    b. What are Uncontrolled components?

    c. Dealing with Controlled vs. Uncontrolled

    d. When to use Controlled vs. Uncontrolled

7. Container vs. Presentational components

    a. What are Container components?

    b. What are Presentational components?

    c. Reasons behind the split

    d. Practice with Container vs. Presentational in simple React

    e. Practice with Container vs. Presentational in React + Redux (or any other state management library)

8. State hoisting

    a. Advantages of hoisting state

    b. Drawbacks and possible gotchas

9. Provider / Consumer pattern

    a. Context API

    b. Using Context API with hooks (useState, useEffect, useReducer)

10. Higher Order Components

    a. What are HOCs?
    b. Practice with HOCs
    c. Example of HOCs in the wild (withRouter, connect, etc.)
    d. When to use HOCs
    e. Drawbacks
11. Functions as child components
    a. What are functions as child components?
    b. Practice with functions as child components
    c. Differences between functions as child components and HOCs
    d. When to use functions as child components
    e. Drawbacks
12. Recap

ABOUT THE TRAINER

Vlad Zelinschi is a pragmatic software architect, lover of the web and caffeine addict. He cares about surrounding himself with thoughtful leaders. He's the CTO of Strongbytes, Google Developer Expert on Web Technologies since October 2016 and a certified trainer. He's part of JSHeroes Community and actively acting as an advisor for a couple of well known Romanian conferences such as Codecamp and NDR.