React JS for Beginners

By Ramesh Fadatare (Java Guides)

My Best Full-Stack Udemy Course:

Spring Boot 3 & React JS: Full-Stack Java Development [2024]

My React JS Complete Tutorial:

React Tutorial

Introduction to React JS

By Ramesh Fadatare (Java Guides)

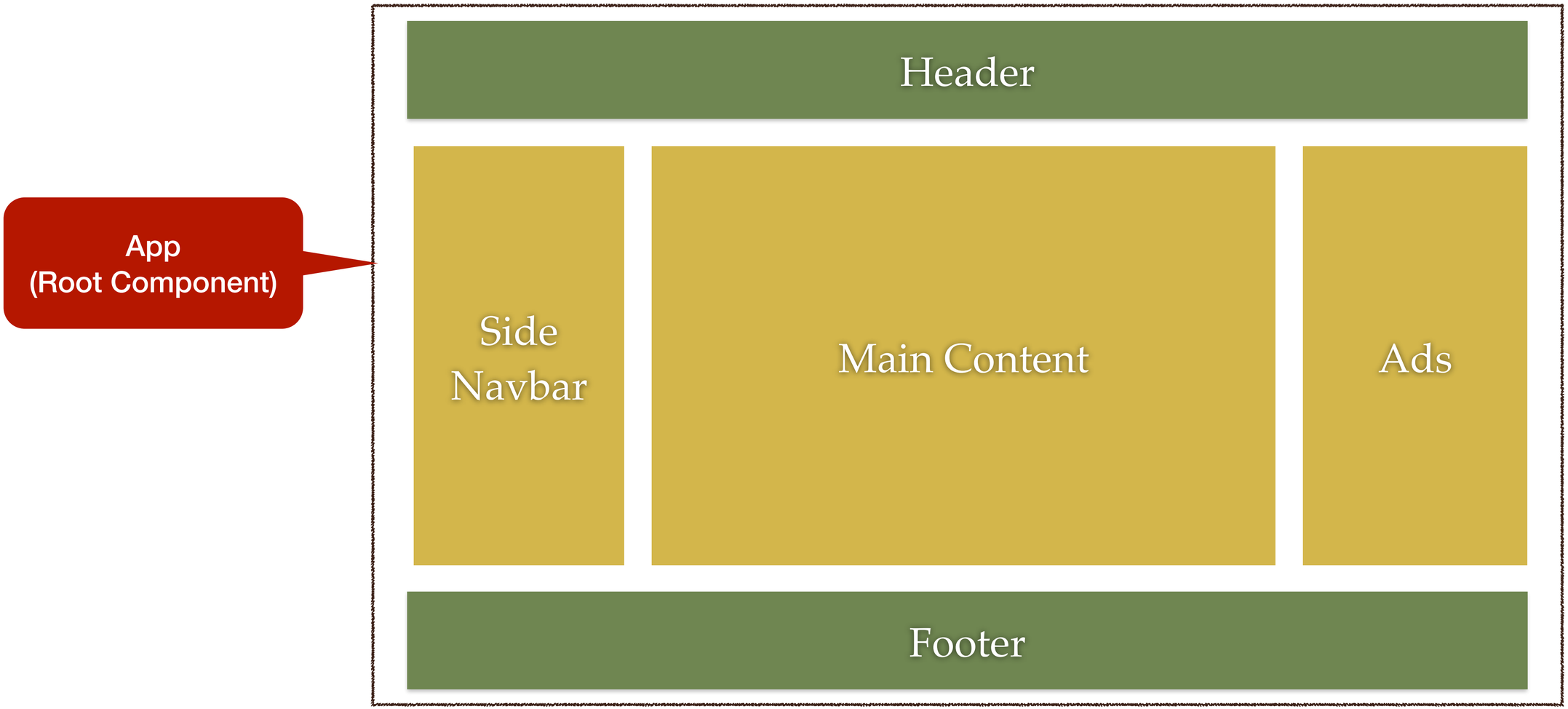
What is React JS?

React JS is an open-source JavaScript library for building user interfaces.

React JS is created by Facebook

React's component-based architecture promotes reusability. Components are reusable building blocks that encapsulate the logic and UI elements of an application. They allow developers to break down complex UI into smaller, manageable parts.

Component-Based Architecture

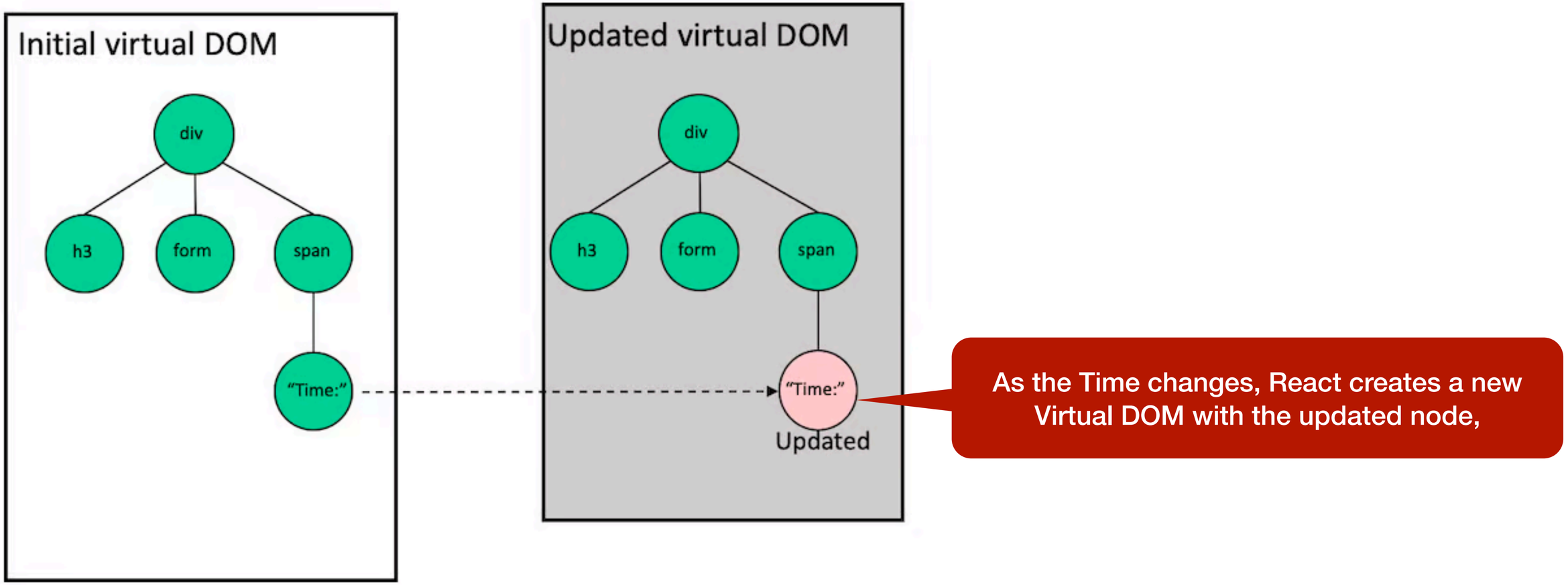


React Virtual DOM

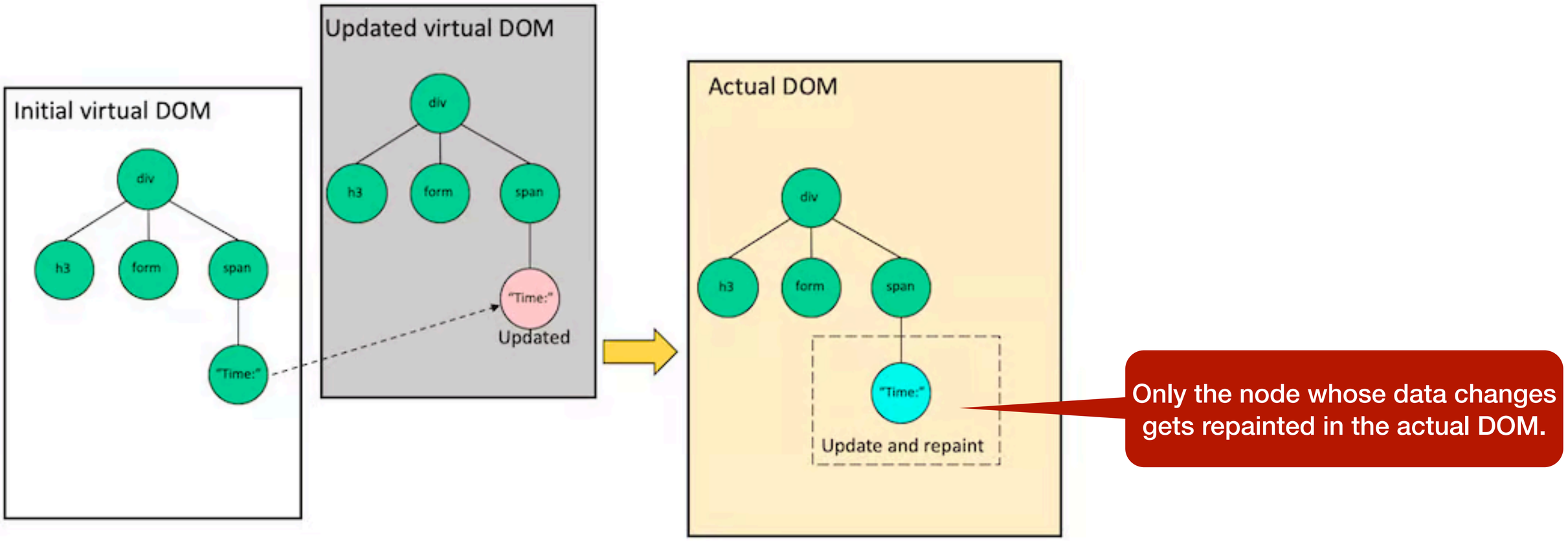
1. ReactJS uses a virtual DOM (Document Object Model) as an abstraction of the real DOM to optimize and efficiently update the user interface.

2. The virtual DOM is a lightweight copy of the actual DOM tree structure. When there are changes in the application state or props, React performs a process called reconciliation to update the virtual DOM efficiently and then applies those changes to the real DOM.

How the React Virtual DOM works



How the React Virtual DOM works



React Real-World Examples

React.js is used by numerous organizations and is the foundation of many popular applications and websites, including **Facebook, Instagram, Airbnb, and WhatsApp**

Create and Set up

React App

By Ramesh Fadatare (Java Guides)

Build Tools for React App

1. Create-React-App
2. Vite JS

Create-React-App

1. Create React App is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.
2. Create React App is built on Webpack and Babel and it is a popular tool that enables developers to quickly set up a React project.
3. It is especially useful for beginners to get started.

Vite JS

1. Vite is a newer build tool that has gained popularity in recent years. It was created to address the limitations of existing build tools, particularly in the development phase.
2. It is built on top of the **Rollup bundler,** which is known for its fast build times.
3. Vite also provides a development server that is optimized for performance.

Create-React-App vs Vite JS

1. Both Create React App and Vite are excellent build tools that provide developers with a solid foundation for building React applications.
2. Create React App is a reliable choice for beginners to learn, while Vite is a better choice for real-world projects that require faster build times.
3. Ultimately, the choice between the two will depend on the project's requirements and the developer's preferences.

Understanding
React App Project Structure

By Ramesh Fadatare (Java Guides)

Components

1. A component is an independent, reusable code block which divides the UI into smaller pieces.
2. Components can nested inside other components
3. Two Types of components in React
    - Functional Components
    - Class Components

Functional Components

1. A functional component is basically a **JavaScript/ES6 arrow function** that returns a React element (JSX).
2. Naming convention: Functional Components always starts with a **capital letter**.
3. Functional Component takes **props** as a parameter if necessary

Class Components

1. Class components are **ES6 classes** that return JSX.
2. Class components (ES6 class) extend the **Component** class in React.
3. Naming convention: Class Components always starts with a **capital** letter.
4. Class Component takes **Props** (in the constructor) if needed
5. Class component must have a **render( )** method for returning JSX

Components Types

JavaScript or ES6 arrow function that returns a React element (JSX).

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

```
const Welcome = (props) => {
  return <h1>Hello, {props.name}</h1>;
}
```

ES6 class that extends **React.Component** and returns a React element (JSX).

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Which Component Type is Recommended to Use

1. React team recommend to use functional components build UI (User Interface).
2. Use ES6 arrow functions to create functional components

Importing and Exporting Components

1. Export component using **export** keyword
2. Import component using **import** keyword
3. Named export and **Default export**
4. We can have multiple **named export** in a single file
5. We can have only one **default export** per file
6. Import Named export using **curly braces**
7. Import default export **without** curly braces

JSX

1. JSX stands for JavaScript XML
2. JSX allows us to write HTML elements with JavaScript code
3. JSX makes it easier to write and add HTML elements in React (no need to use **createElement()**)
4. JSX will be converted to JavaScript on browser using a transpiler - babel.js.
5. Babel is a library which transpiles JSX to pure JavaScript that the browser understands.

JSX Rules in React

1. Component should return single React element (div / section / article / fragment)
2. To write HTML on multiple lines, put the HTML inside parentheses
3. Use camelCase for HTML Attribute
    - To specify CSS classes for an element, use the **className** attribute instead of the **class** attribute. This is because **class** is a reserved keyword in JavaScript.
    - Event handlers in JSX are specified using camelCase syntax. For example, **onClick** instead of **onclick** or **onChange** instead of **onchange**.
4. **Expressions within Curly Braces:** You can embed JavaScript expressions within JSX elements by wrapping them in curly braces {}.
5. JSX follows XML rules, and therefore HTML elements must be properly closed. If an element doesn't have any children, you can use a self-closing tag.

Props (properties)

1. The **props** is a special keyword in React that stands for **properties** and is being used to pass data from one component to another and mostly from parent component to child component.
2. We can say **props** is a data carrier or a means to transport data.
3. React **props** is an object which you get instantly when you create a React component.

Destructuring props

1. Destructuring was introduced in ES6. It's a JavaScript feature that allows us to extract multiple pieces of data from an array or object and assign them to their own variables.
2. In React, destructuring props improve code readability.
3. Two ways to destructure props in functional component
    - The first way is destructuring it in the function parameter itself
    - The second way is destructuring props in the function body

State and setState in Class Components

1. The **state** is a built-in object in React class components. In the state object, we store property values that belong to the component.
2. When the **state** object changes, the component re-renders.
3. We use **setState()** method to change the state object in a class component

useState Hook in Functional Components

1. **Question:** Only class components has in-built state object then how to define state variables in functional components.
2. The **useState** is a Hook (function) that allows you to have state variables in functional components.
3. To use hooks, first we should import the **useState** hooks from react. The **useState** is a function which takes one argument and returns a current state and function that lets you update it.

Event Handling

1. In React, event handling allows you to handle user interactions, such as button clicks, form submissions, or mouse movements, within your application.

2. Event handling in React follows a similar pattern to standard JavaScript event handling but with some differences:

    - React events are written in camelCase syntax: **onClick** instead of **onclick**.

    - React event handlers are written inside curly braces: **onClick={shoot}** instead of **onClick="shoot()"**

Conditional Rendering

1. Conditional rendering in React allows you to render different content or components based on certain conditions or state values. It helps you dynamically control what gets displayed in your UI based on specific conditions.

2. Three different ways to implement conditional rendering:

    1. Conditional Rendering using If and Else statement

    2. Conditional Rendering using Ternary Operator

    3. Conditional Rendering using && Operator (Short Circuit Operator)

Form Handling in React

1. **Form Structure:** Define the structure of your form using HTML elements - <form>, <input>, <select>, etc.
2. **State Setup:** Use the **useState** hook to create state variables to store the form data. Typically, each form element corresponds to a state variable that will hold its value.
3. **Event Handling:** Attach event listeners to the form elements to handle changes in their values. In React, you typically use the onChange event to capture user input. When an input value changes, the event handler should update the corresponding state variable.
4. **Submit Handling:** Attach an event listener to the form's onSubmit event. When the form is submitted (e.g., by clicking a submit button), the event handler is triggered.

Form Handling - Development Steps

1. Use the **useState** hook to create state variables to store the form data. Typically, each form element corresponds to a state variable that will hold its value.

2. Design the form using HTML elements <form>, <input>, <select>, etc.

3. Attach event listeners to the form elements to handle changes in their values. In React, you typically use the **onChange** event to capture user input.

4. Attach an event listener to the form's **onSubmit** event. When the form is submitted (e.g., by clicking a submit button), the event handler is triggered.

My Best Full-Stack Udemy Course:

Spring Boot 3 & React JS: Full-Stack Java Development [2024]

My React JS Complete Tutorial:

React Tutorial