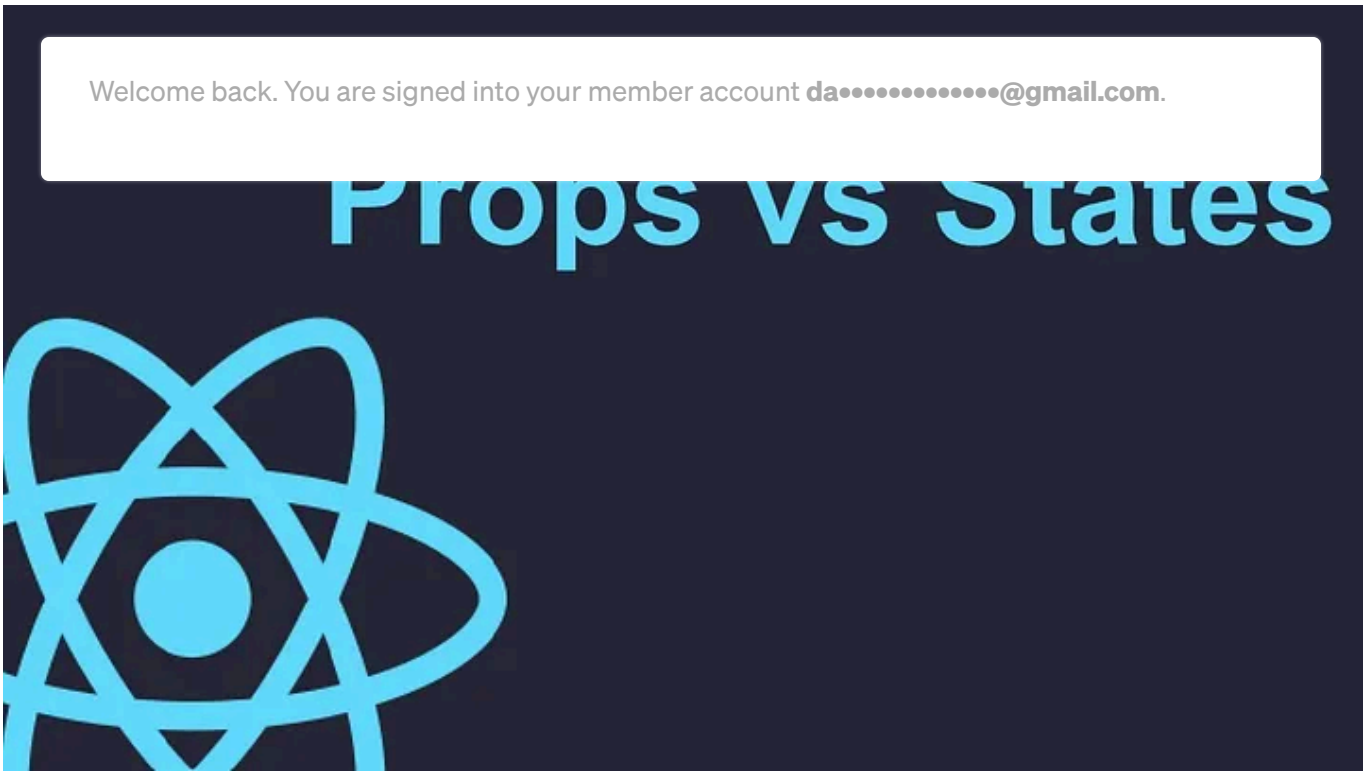
React JS: A Comprehensive Guide to State and Props

PAVAN BIRARI · Follow
4 min read · Feb 1, 2024

> *React JS is a popular JavaScript library used for building user interfaces. It is known for its simplicity, flexibility, and performance. In this guide, we will explore two important concepts in React JS: state and props.*



Understanding State in React JS

State is a built-in object in React JS that allows components to store and manage their own data. It is a way for components to keep track of their internal state and update their UI based on changes to that state.

What is state ?

State is an object that holds information about the component's current state. It can be initialized in the constructor method of a component and updated using the `setState()` method. When the state of a component changes, React will automatically re-render the component to reflect the new state.

How does state work ?

State works by allowing components to store and manage their own data. When a component's state changes, React will automatically re-render the

component to reflect the new state. This makes it easy to create dynamic and in

How to use state in your code

To use state in your code, you first need to initialize it in the constructor method of your component. You can then update the state using the `setState()` method. Here's an example

```
class MyComponent extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }

  handleClick() {
    this.setState({
      count: this.state.count + 1
    });
  }

  render() {
    return (
      <div>
        <p>You clicked {this.state.count} times</p>
        <button onClick={() => this.handleClick()}>
          Click me
        </button>
      </div>
    );
  }
}
```

In this example, we're initializing the state of our component to an object with a single property called `count`. We're then updating the state whenever

the user clicks the button by calling the `handleClick()` method and using the `se`

Examples of state in action

1. **Toggle a button:** You can use state to toggle the state of a button between "on" and "off" when it's clicked.
2. **Display a counter:** You can use state to keep track of how many times a button has been clicked and display the count to the user.
3. **Show or hide content:** You can use state to show or hide content based on user interaction.

Understanding Props in React JS

What are props ?

Props are read-only values that can be passed across components in order to display relevant data in your React apps. They are similar to function arguments in JavaScript and attributes in HTML .

How do you pass data with props ?

To pass data with props, you can use the same syntax as HTML attributes. For example, you can pass an object and a number as props to a child component like

```
<ChildComponent person={{ name: 'John Doe', age: 30 }} count={10} />
```

In the above example, we're passing an object with two properties (`name` and `age`).

Examples of props in action

1. **Displaying data:** You can use props to display data in your React components. For example, you can pass an object with user information as props to a `UserProfile` component and display the user's name, age, and profile picture.
2. **Passing functions:** You can pass functions as props to child components to allow them to communicate with their parent components. For example, you can pass a function that updates the state of a parent component to a child component and call it when a button is clicked.
3. **Conditional rendering:** You can use props to conditionally render content in your React components. For example, you can pass a boolean value as props to a `ShowHide` component and render different content based on whether the value is true or false.

Differences Between State and Props

State is an object that holds information about the component's current state. It can be initialized in the constructor method of a component and updated using the `setState()` method. When the state of a component changes, React will automatically re-render the component to reflect the new state. State is local to the component and can only be accessed and modified within that component.

Props are read-only values that can be passed from one component to another. They are used to pass data between components and allow you to create consistent interfaces across the component hierarchy. Props are

owned by the parent component and passed down to child components.

Th

Here are some key differences between state and props:

1. **Ownership:** State is owned and managed within the component itself, while props are owned by the parent component and passed down to child components.
2. **Mutability:** State is mutable and can be changed using the `setState()` method, while props are immutable and cannot be changed by the child component.
3. **Access:** State is local to the component and can only be accessed and modified within that component, while props can be accessed by the child component but cannot be modified.
4. **Usage:** State is used to manage data that is local to the component, while props are used to pass data between components.

State and props are two important concepts in React JS. State is local to the component, while props are passed from parent to child components.

> 🧡*Thank you for reading. I hope this will help you on your journey !* 💖
>
> *Thank you for being a part of the community! Before you go…..*
>
> ***Be sure to clap and follow the writer PAVAN BIRARI will miotiviate us to write more ..*** 💛

Written by PAVAN BIRARI

Follow

9 Followers

Software Developer. Experties in | ReactJs | Javascript | Redux | HTML | CSS | Problem Solving

More from PAVAN BIRARI

PAVAN BIRARI

Mastering Promises in JavaScript

A Comprehensive Guide for 2024

PAVAN BIRARI

Mastering HTML Forms: Best Practices and Techniques

In this blog we are discussing about the best pracitices and techniques of html Form...

Jan

PAVAN BIRARI

Mastering JSX: A Comprehensive Guide to React's Syntax Extension

Welcome to the world of JSX, a powerful syntax extension for JavaScript that has...

Jan 28 5

PAVAN BIRARI

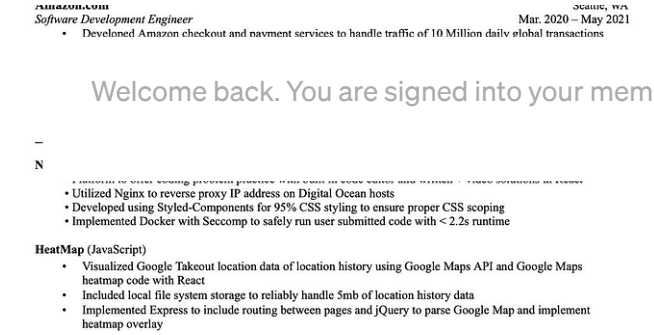
React.js Folder Structure: A Comprehensive Guide

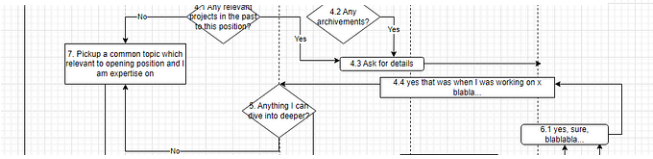
Your One-Stop Series for Learning and Implementing React.js in Real-World...

Jan 22 6

See all from PAVAN BIRARI

Recommended from Medium





Alexander Nguyen in Level Up Coding

The resume that got a software engineer a $300,000 job at Google.

1-page. Well-formatted.

Jun 1 21K 392

LORY

Today I Interviewed for a Lead Front-End Role

And They Asked Me a Couple of Tough Questions

Aug 3 805 12

Lists

Growth Marketing

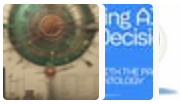
11 stories · 222 saves

Stories to Help You Grow as a Software Developer

19 stories · 1354 saves

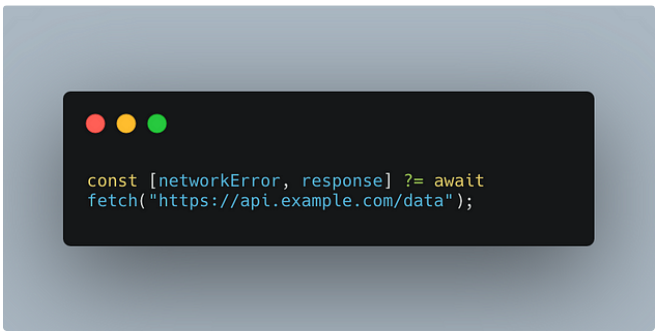
General Coding Knowledge

20 stories · 1566 saves



data science and AI

40 stories · 241 saves



Selcuk Ozdemir in JavaScript in Plain English

Josh Lee

Say Goodbye to Try/Catch with This Ne

Int

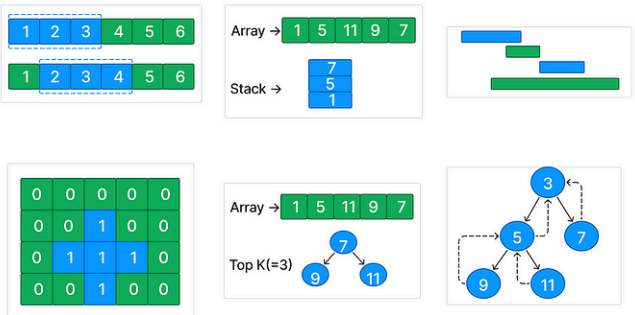
Error Handling Hero

Sep 3 1.6K 35

Factory Functions in Javascript:

years ago, I primarily worked with Ruby. And...

Mar 21 10 1



Ashish Pratap Singh in AlgoMaster.io

LeetCode was HARD until I Learned these 15 Patterns

I have solved more than 1500 LeetCode problems, and if there is one thing I have...

Aug 23 1.6K 16

Thiraphat Phutson

Best Practices for Structuring a React TypeScript Project

Structuring a React TypeScript project effectively is crucial for maintaining code...

May 20 20 1

See more recommendations