ReactJs-

Cmd-

npx create-react-app project name

cd projectname

npm start

import './App.css';

import {useState} from 'react'

import React from 'react';

function App() {

  const[count,setCount] =useState(0)

  function updateCount(){

    setCount(count+1)

  }

return (

<>

   <p>

    Learning React....

   </p>

   <h1>button click no of times:{count}</h1>

   <button

   onClick={updateCount}

   >Click</button>

   </>

  );

}

export default App;

UseEffect-

**1.Empty Dependency Array ([])**:

* The effect runs only once, after the initial render.
* It does not run again on subsequent renders, unless the component is unmounted and re-mounted.
* Useful for initialization logic (e.g., fetching data, setting up subscriptions).

 useEffect(()=>{

    console.log("Component created:")

  },[])

**2.No Dependency Array**:

 The effect runs after every render.

 Useful for effects that need to run on every update.

Ex-

useEffect(() => {

console.log("This runs after every render.");

});

**3.Dependency Array with Specific Variables**:

* The effect runs only when any of the variables in the array change.
* Useful for effects that depend on specific state or props.

Ex-

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

useEffect(() => {

console.log("This runs when 'count' changes:", count);

}, [count]);

Q-Why to learn React?

->makes easy to manage and build complex front end.

->Build UI, Trendy

* When should I learn React?

->After mastering JS

1.React is a Library

Q-What is framework and library?

Framework-follow a rules an naming convention.

Library- cool dude- (freedom jada hai).

2.props means properties

If we create object we can pass {key:value}

3.topics to learn

->Core of react(state Or UI manipulation ,JSX)

->Components Reusability

->Reusing of components(Props)

->how to propagate chnge(hooks)

4.Additional Addon to React

->Router(React don’t have Router)

->State Management (React don’t have Router)

->Redux, redux toolkit, context Api

->class based component

->BAAS Apps

* Package.json

1.script-

Is use to get ready for production.

2.test- to test run cases

3.eject-

* First create project

->Npx-full form (node pckg extecuter)

-> create-react-app –this is a software

* What is React:  React is a javascript library for building user interfaces. React is a popular, declarative, component based state-driven javascript library for building user interfaces. Created 2011 by jordan walke  facebook.

1. Component: Components are the building blocks of user interfaces in react.
2. State driven means always keeping the sync ui with data.

**Setting up environment:**

* First Install VS Code
* Install Node JS 18 /20/22
* In vs code install ESLint extension for finding errors and some best practices.
* Install Prettier code formatter extension in vs code .
* Install one monokai theme extension.
* Install Material Icon theme extension.
* Change vs code settings auto save to onFocusChange.
* Open terminal and check the node version using ‘node -v’ command.

**Two ways for creating App:**

**1. create-react-app:**: a complete starter kit for React applicationsthat was developed many years agoin order to make it really easy for developers to scaffold new React apps.So an app created with create-react-app “projectName”

Steps-Cmd=>

1.npx create-react-app projectName(It takes more time)

**2: Using Vite:**

* Now, Vite is actually quite different from Create-React-App as it's basically simply a modern build tool.So a bit like a modern webpack but which happens to also contain a starter template for setting up brand new React applications. However, in a React app created with Vite, you will have to manually set up many important developer tools, such as ESLint, Prettier, a testing library, and so on.

Vitecmd-(it take less time)

cmdLink- https://vitejs.dev/guide/

Cmd-

1) npm create vite@latest

2)project name

3)react

4)javascript

Cd projectName

5)npm install

6)npm run dev(In pckg.json->script was given

React first program-

File-App.jsx

Import Chai from “./chai”

Function App(){

return(

<Chai/>

Note:.jsx file rule ---(return only one elements).

Solution🡪

<>Or<div>

<Chai/>

<h1>chai aur react></h1>

<p>test para</p>

</> or</div>

)

}

Export default App

File-Chai.jsx

Function Chai(){

return({

<h3> chai is ready..</h3>

)

}

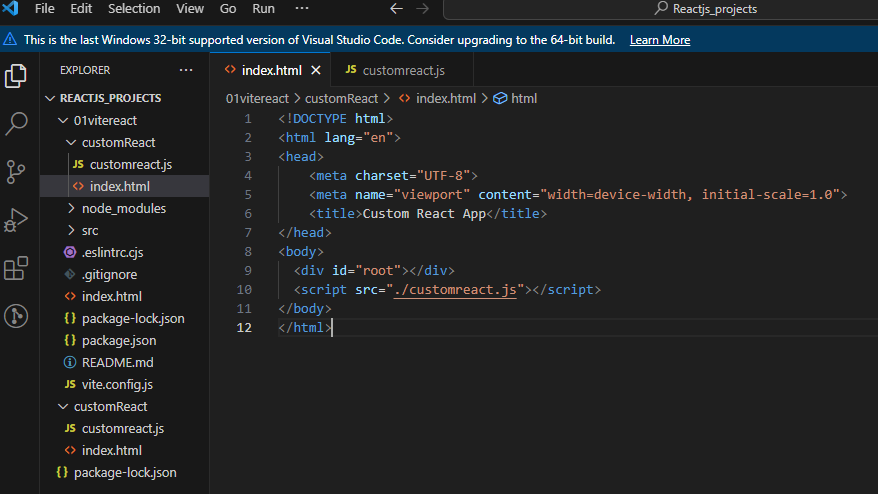
Export default Chai

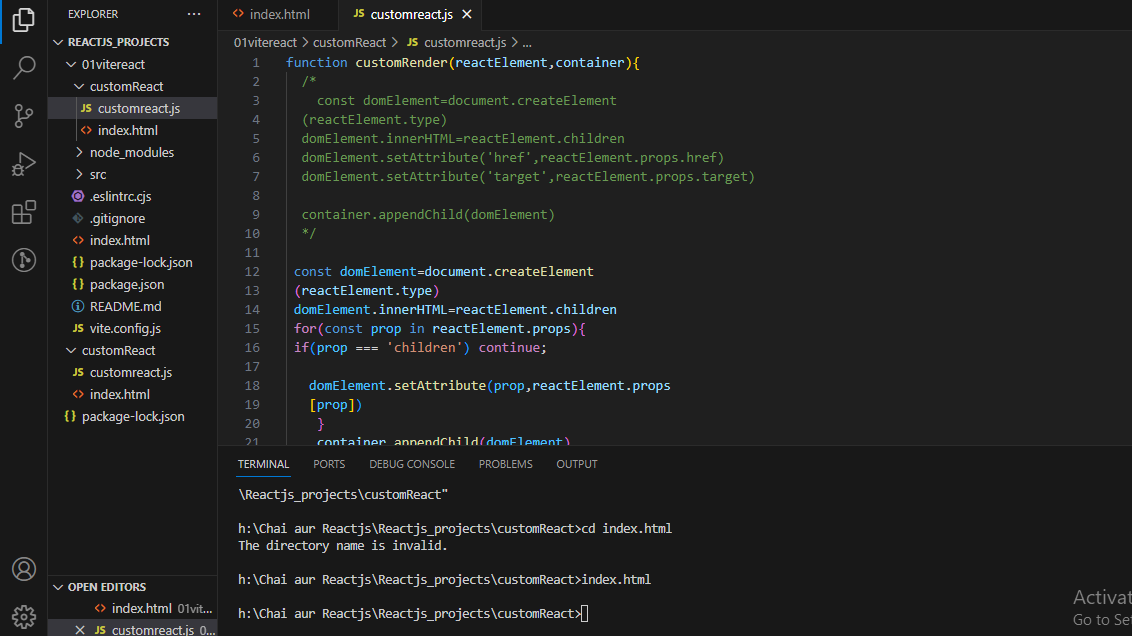
Note:

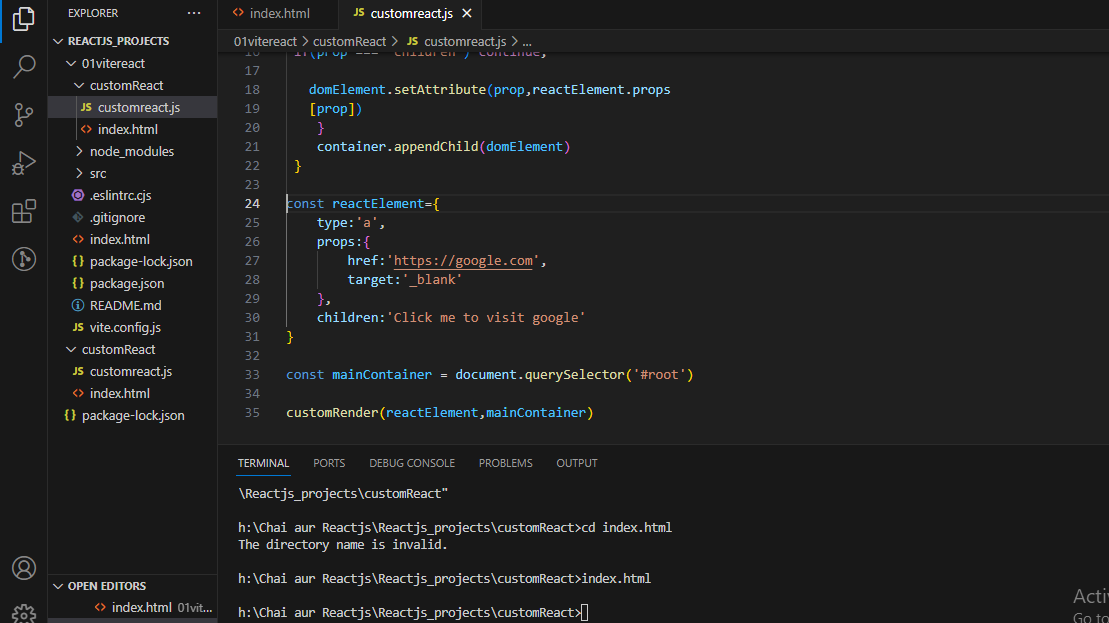
1.browser me use krte to ReactDOM use krte hai.

2.mobile me use krte hai to React Native use krte hai.

Program2- **Create your own react library and JSX**

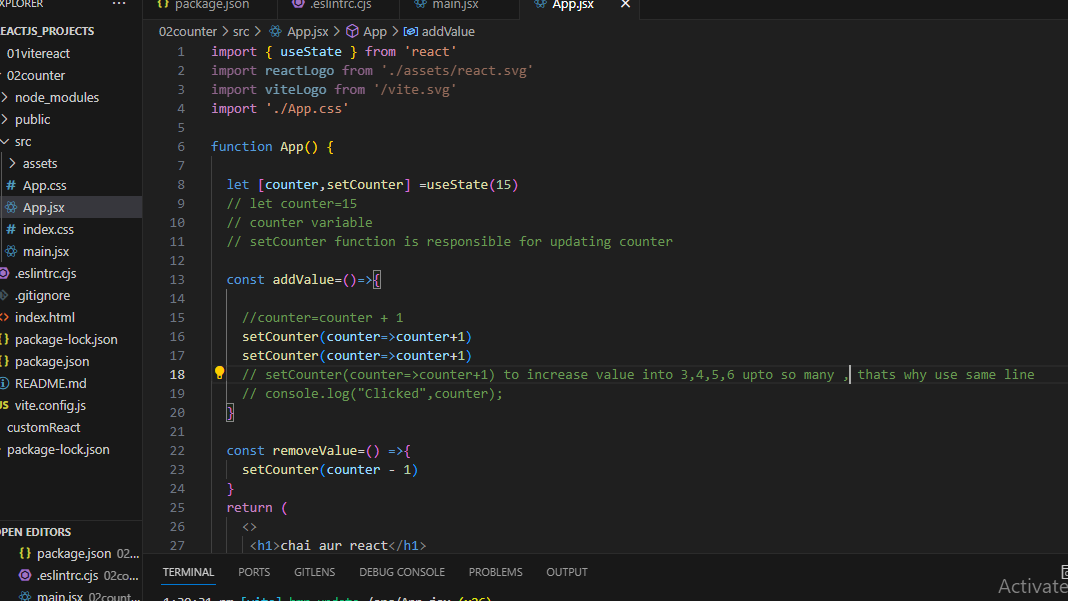






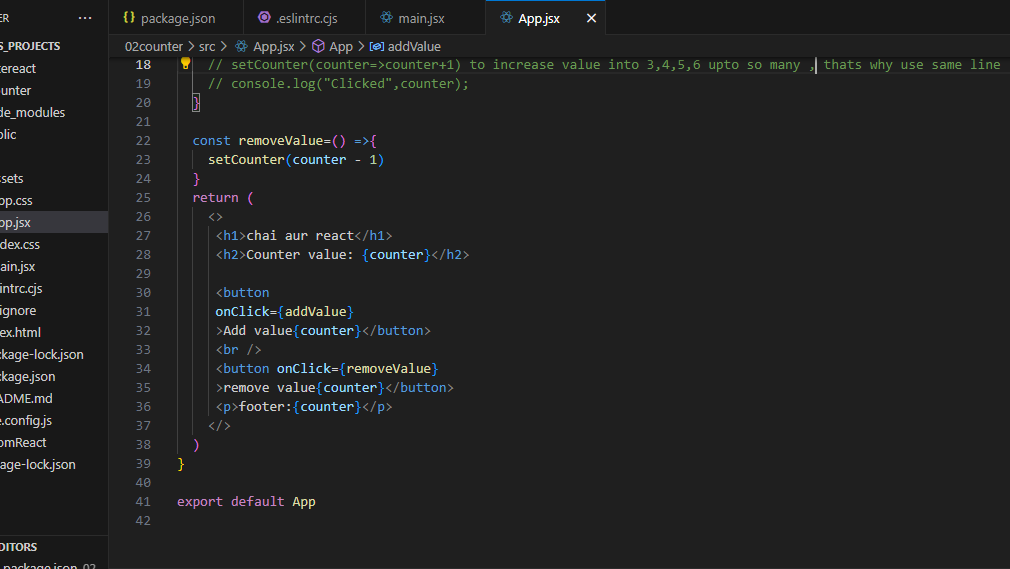
Date-11 july

Why you need hooks and project



Output-





Virtual Dom ,Fibre and reconciliation-

## Virtual Dom-

## ****What is DOM ?****

DOM stands for ‘Document Object Model’. In simple terms, it is a structured representation of the HTML elements that are present in a webpage or web app. DOM represents the entire UI of your application. The DOM is represented as a tree data structure. It contains a node for each UI element present in the web document. It is very useful as it allows web developers to modify content through JavaScript, also it being in structured format helps a lot as we can choose specific targets and all the code becomes much easier to work with.

## Disadvantages of real DOM :

Every time the DOM gets updated, the updated element and its children have to be rendered again to update the UI of our page. For this, each time there is a component update, the DOM needs to be updated and the UI components have to be re-rendered.  
**Example:**

// Simple getElementById() method  
document.getElementById('some-id').innerValue = 'updated value';

**Virtual DOM**

React uses Virtual DOM exists which is like a lightweight copy of the actual DOM(a virtual representation of the DOM). So for every object that exists in the original DOM, there is an object for that in React Virtual DOM. It is exactly the same, but it does not have the power to directly change the layout of the document.

## Differences between Virtual DOM and Real DOM

| **Virtual DOM** | **Real DOM** |
| --- | --- |
| It is a lightweight copy of the original DOM | It is a tree representation of HTML elements |
| It is maintained by JavaScript libraries | It is maintained by the browser after parsing HTML elements |
| After manipulation it only re-renders changed elements | After manipulation, it re-render the entire DOM |
| Updates are lightweight | Updates are heavyweight |
| Performance is fast and UX is optimised | Performance is slow and the UX quality is low |
| Highly efficient as it performs batch updates | Less efficient due to re-rendering of DOM after each update |

React fibre-link(<https://github.com/acdlite/react-fiber-architecture>)

->React Fiber is an ongoing reimplementation of React's core algorithm. It is the culmination of over two years of research by the React team.

The goal of React Fiber is to increase its suitability for areas like animation, layout, and gestures. Its headline feature is **incremental rendering**: the ability to split rendering work into chunks and spread it out over multiple frames.

Other key features include the ability to pause, abort, or reuse work as new updates come in; the ability to assign priority to different types of updates; and new concurrency primitives.

### What is reconciliation?

*reconciliation*

The algorithm React uses to diff one tree with another to determine which parts need to be changed.

## What is a fiber?

->We're about to discuss the heart of React Fiber's architecture. Fibers are a much lower-level abstraction than application developers typically think about.

We've established that a primary goal of Fiber is to enable React to take advantage of scheduling. Specifically, we need to be able to

* pause work and come back to it later.
* assign priority to different types of work.
* reuse previously completed work.
* abort work if it's no longer needed.

# Tailwind and Props in reactjs with Example

Props:

* React Props are like function arguments in JavaScript *and* attributes in HTML.
* To send props into a component, use the same syntax as HTML attributes:
* In the props data only flows down the tree from parent to child. It does not flows between the siblings.
* Props are used to pass data from a parent component to a child component. They are passed as attributes to the child component when it is declared in the parent component’s [JSX](https://www.geeksforgeeks.org/reactjs-jsx-introduction/).
* Props Example-

//Parent Component

function ParentComponent(){

const msg=”hello sakshi”;

return <ChildComponent greeting={msg} />;

}

//Child Component

function ChildComponent(props){

return <p>{props.greeting}</p>;

}

1.Link-for tailwind cmds—(<https://tailwindcss.com/> and <https://tailwindcss.com/docs/guides/vite>)

2.Then run this cmd-

npm install -D tailwindcss postcss autoprefixer

npx tailwindcss init -p

3.Then in tailwind.config.js file we change content block.

content: [

"./index.html",

"./src/\*\*/\*.{js,ts,jsx,tsx}",

],

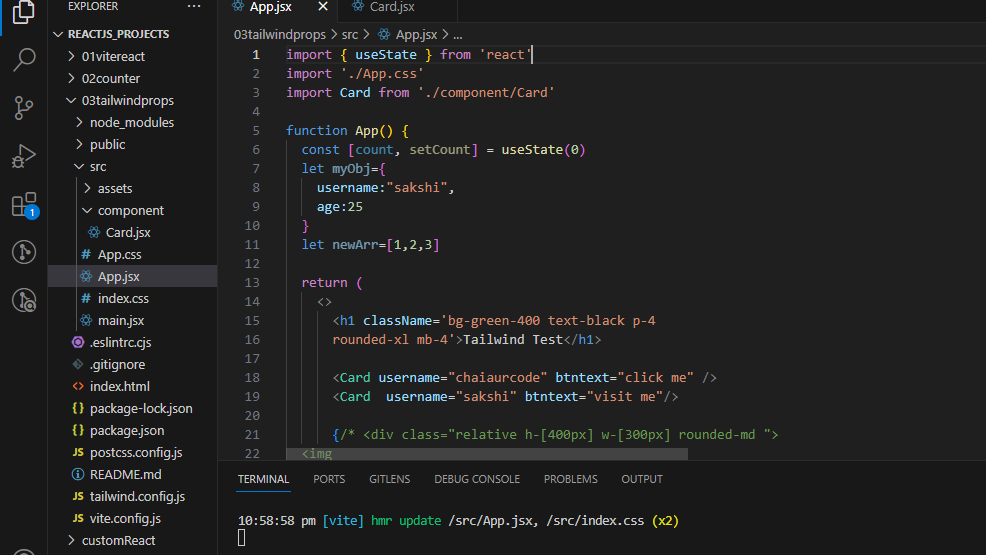
4.Then change index.css code to write

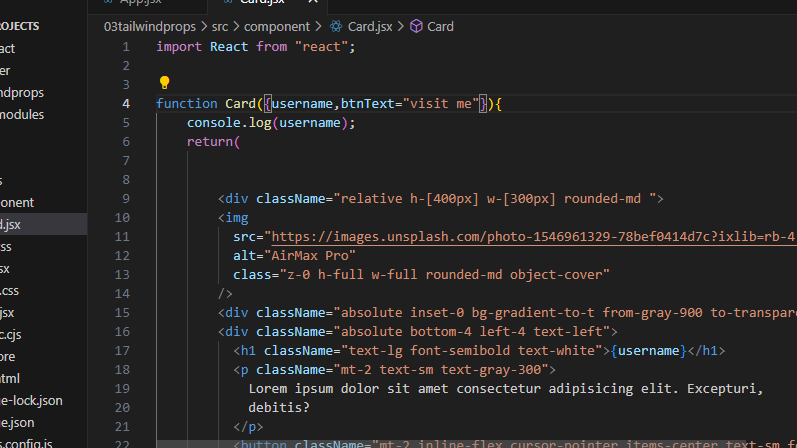
Index.css file

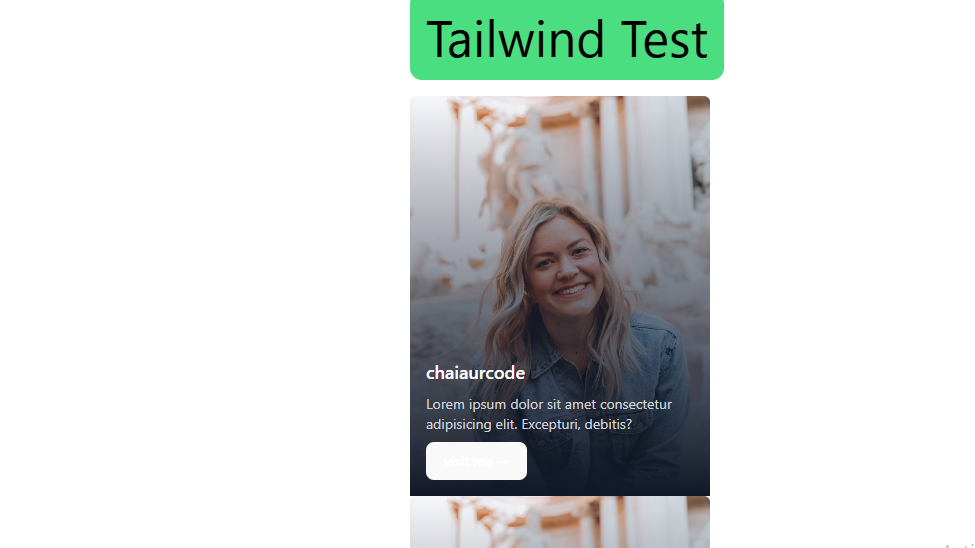
@tailwind base;

@tailwind components;

@tailwind utilities;







# Building a react project | bgChanger

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1052).png

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1053).png

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1054).png

# useEffect, useRef and useCallback with 1 project-

# Vs code-Ex 05passwordGenerator

# 

# 12 july

# Custom hooks in react | currency Project

# ->Hooks are reusable functions.

# ->When you have component logic that needs to be used by multiple components, we can extract that logic to a custom Hook.\

# -> Custom Hooks start with "use". Example: useFetch

# Program-06currencyConvertor

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1057).png

# Context API crash course with 2 projects

# Website for documentation->context api react.dev-

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1059).png

# 15 july

# Context api with local storage | project

# Project-10todocustomLocal

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1061).png

# 16 july

Redux toolkit crash course-

The **Redux Toolkit** package is intended to be the standard way to write [Redux](https://redux.js.org/" \t "_blank) logic. It was originally created to help address three common concerns about Redux:

Redux->core libraray

React Redux->implementation of redux for wireing(react redux shortly use for communication of both)

12MegaBlock—Website for more info(https://vitejs.dev/guide/env-and-mode.html)

1.create project

2.pckg install-

-npm i @reduxjs/toolkit

-react-redux-router-dom appwrite

-@tinymce/tinymce-react

-html-react-parser

-react-hook-form

Appwrite backend for react project-

-Appwrite website used for file conf.js data insertion

-For documentation---( <https://appwrite.io/docs/references/cloud/client-web/databases#databasesUpdateDocument>)

17 july

Appwrite Database ,file upload and custom queries

# 18 july

# How to use React hook form in production

# MegaBlog project-

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1064).png

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1063).png

# C:\Users\Aakash\Pictures\Screenshots\Screenshot (1065).png

## What is ES6?

ES6 stands for ECMAScript 6.

ECMAScript was created to standardize JavaScript, and ES6 is the 6th version of ECMAScript, it was published in 2015, and is also known as ECMAScript 2015.

## Why Should I Learn ES6?

React uses ES6, and you should be familiar with some of the new features like:

* [Classes](https://www.w3schools.com/react/react_es6_classes.asp)
* [Arrow Functions](https://www.w3schools.com/react/react_es6_arrow.asp)
* [Variables](https://www.w3schools.com/react/react_es6_variables.asp) (let, const, var)
* [Array Methods](https://www.w3schools.com/react/react_es6_array_methods.asp) like .map()
* [Destructuring](https://www.w3schools.com/react/react_es6_destructuring.asp)
* [Modules](https://www.w3schools.com/react/react_es6_modules.asp)
* [Ternary Operator](https://www.w3schools.com/react/react_es6_ternary.asp)
* [Spread Operator](https://www.w3schools.com/react/react_es6_spread.asp)

## The render Method

The render() method is then called to define the React component that should be rendered.

const container = document.getElementById('root');

const root = ReactDOM.createRoot(container);

root.render(<p>Hello</p>);

## What is JSX?

JSX stands for JavaScript XML.

JSX allows us to write HTML in React.

JSX makes it easier to write and add HTML in React.

## React Props

React Props are like function arguments in JavaScript and attributes in HTML.

To send props into a component, use the same syntax as HTML attributes:

### Example -

Add a "brand" attribute to the Car element:

const myElement = <Car brand="Ford" />;

### Example-

The component receives the argument as a props object:

Use the brand attribute in the component:

function Car(props) {

return <h2>I am a { props.brand }!</h2>;

# React Events

Just like HTML DOM events, React can perform actions based on user events.

React has the same events as HTML: click, change, mouseover etc.

onClick instead of onclick.

# React Hooks

Hooks were added to React in version 16.8.

Hooks allow us to "hook" into React features such as state and lifecycle methods.

You must import Hooks from react.

the useState Hook to keep track of the application state.

## Hook Rules

There are 3 rules for hooks:

* Hooks can only be called inside React function components.
* Hooks can only be called at the top level of a component.
* Hooks cannot be conditional

1.usestate

2.useEffect

3.useContext

4.useRef