React.js Notes Complete Guide for Beginners to Advanced

Hey Coder! Welcome to the world of React.js. Yeh notes specially aapke liye banaye gaye hain, taaki aap React ko easily aur fun tareeke se samajh sakein. **Sochna band, coding shuru! React bilkul Lego blocks jaisa hai; chote-chote pieces (components) jodkar aap kuch bhi bada aur amazing bana sakte ho.** Chalo, React ki duniya mein ek exciting journey shuru karte hain!

Created with love by **Ashwini Kumar** (via Gemini) September 15, 2025

Contents

Chapter 1

Getting Started with React (React ka Introduction)

1.1 What is React? (Yeh React hai kya?)

React is a **JavaScript library** for building user interfaces (UIs).

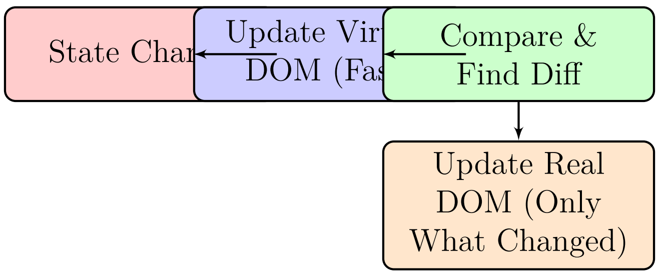
- **Main Kaam**: **React ka main kaam hai UI ko chote-chote, independent pieces mein todna, jinhe hum ‘components‘ kehte hain.**
- **Analogy**: **Socho aapko ek car banani hai. Poori car ek saath banane se behtar hai ki aap pehle wheels, engine, seats, aur steering alag-alag banao aur phir unhe jod do. Yeh wheels, engine, seats hi ‘components‘ hain.**

1.2 Why React? The "Virtual DOM" Magic

React's secret to speed is the **Virtual DOM**.

- **Kya hai?**: **Normally, jab aap web page par chota sa bhi change karte ho, poora page (Real DOM) refresh hota hai. Yeh bahut slow hai.**
- **Kaise kaam karta hai?**: **React Real DOM ki ek copy (Virtual DOM) memory mein rakhta hai. Jab koi change hota hai, React pehle Virtual DOM ko update karta hai. Phir woh compare karta hai ki Real DOM aur naye Virtual DOM mein kya antar hai. React sirf uss ‘antar‘ (difference) ko Real DOM mein update karta hai.** This process is called **reconciliation**.

Diagram: Virtual DOM vs Real DOM



1.3 Setting Up Your First React App (Vite ke saath!)

We use **Vite** because it's blazing fast.

1. **Run this command in your terminal:**

```
npm create vite@latest my-first-react-app --template react

```

2. **Go to the new directory and install packages:**

```
cd my-first-react-app
npm install

```

3. **Start the development server:**

```
npm run dev

```

Bas! Aapka React app ‘http://localhost:5173‘ par live hai.

Practice Zone
Sharpen your setup skills by creating a new Vite project with a different template, like TypeScript: `-template react-ts`. Check out the Vite docs for more: Vite Official Guide.

Your Challenge
Create a new React project using Vite. Open ‘src/App.jsx‘ and change the text inside the ‘<h1>‘ tag to "My First React App". Save and see the change instantly in your browser!

Quick MCQs

1. What is React?
   a) A JavaScript Framework
   a) **A JavaScript Library**
   a) A CSS Framework
   a) A Server-side language
2. The core concept of React is to break the UI into:
   b) **Components**
   b) Functions
   b) Files
   b) APIs
3. What makes React fast?

c) Real DOM
c) Shadow DOM
c) **Virtual DOM**
c) Document API

4. Which command starts the development server in a Vite project?

d) npm start
d) **npm run dev**
d) npm build
d) node start

5. React was developed by which company?

e) Google
e) Microsoft
e) **Facebook (Meta)**
e) Apple

Chapter 2

Core Concepts (React ke Funde)

2.1 JSX: JavaScript aur HTML ka Perfect Mix

JSX stands for JavaScript XML. It lets you write HTML-like syntax inside your JavaScript code.

- **Kyun use hota hai?**: **Isse UI ka structure likhna bahut natural lagta hai. Code dekh kar hi samajh aa jaata hai ki page kaisa dikhega.**
- **Kaise kaam karta hai?**: Browser JSX nahi samajhta. A tool called **Babel** converts it into regular ‘React.createElement()‘ calls that JavaScript understands.

Example:

```
// This is JSX
const userName = "Ashwini";
const element = <h1>Hello, {userName}! Welcome to React.</h1>;

// Babel converts it to this behind the scenes:
const element = React.createElement('h1', null, 'Hello, ',
  userName, '! Welcome to React.');
```

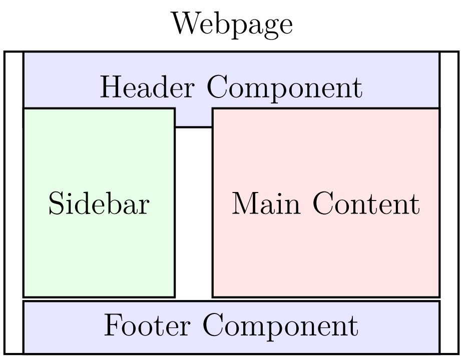
Tip: In JSX, use ‘className‘ instead of ‘class‘ for CSS classes, because ‘class‘ is a reserved keyword in JavaScript.

2.2 Components: The Building Blocks

In React, everything is a component. We primarily use **Functional Components**.

- **Functional Component**: A simple JavaScript function that accepts ‘props‘ as an argument and returns a React element (JSX).

Diagram: A Webpage as Components



2.3 Props: Parent se Child ko Data Transfer

Props (properties) are used to pass data from a parent component to a child component.

- **Analogy**: **Props bilkul ek gift ki tarah hain jo parents (parent component) apne bachchon (child component) ko dete hain. Bachcha gift ko use kar sakta hai, par use badal nahi sakta. Props are read-only!**

Example:

```
// Parent Component
function App() {
  return (
    <div>
      <UserProfile name="Ashwini" age={25} location="Bhopal" />
      <UserProfile name="Rohan" age={22} location="Mumbai" />
    </div>
  );
}

// Child Component
function UserProfile(props) {
  // Accessing data using props
  return (
    <div className="user-card">
      <h2>Name: {props.name}</h2>
      <p>Age: {props.age}</p>
      <p>Location: {props.location}</p>
    </div>
  );
}
```

2.4 State: Component ki apni Memory

If a component needs to remember something that can change over time (like a user's input or a button click count), it uses **state**.

- **Analogy**: **State ek component ki personal diary jaisa hai. Woh usmein kuch bhi likh sakta hai aur badal sakta hai. Jab bhi diary mein kuch**

naya likha jaata hai (state changes), component uss change ko UI par dikha deta hai (re-renders).

Example (using ‘useState‘ Hook):

```
import { useState } from 'react';

function LightSwitch() {
  // 'isOn' is the state variable. 'setIsOn' is the function to
   change it.
  const [isOn, setIsOn] = useState(false); // Initial state is '
   false' (off)

  const flipSwitch = () => {
    setIsOn(!isOn); // Toggle the state from true to false and
   vice-versa
  };

  return (
    <div>
      <p>The light is {isOn ? 'ON' : 'OFF'}</p>
      <button onClick={flipSwitch}>Flip Switch</button>
    </div>
  );
}
```

Practice Zone
Explore more about JSX on the official React docs: Writing Markup with JSX. And learn about State here: State: A Component’s Memory.

Your Challenge
Create a ‘Product‘ component that accepts ‘name‘ and ‘price‘ as props and displays them. Then, in your ‘App‘ component, render three different ‘Product‘ components with unique names and prices.

Quick MCQs

1. JSX is converted into what by Babel?
   a) Regular HTML
   a) CSS
   a) **‘React.createElement()‘ calls**
   a) JSON objects
2. How is data passed from a parent to a child component?
   b) Using state
   b) **Using props**

   b) Using events
   b) Using refs

3. What happens when a component's state changes?
   c) The entire application reloads.
   c) Only the parent component re-renders.
   c) Nothing happens.
   c) **The component and its children re-render.**

4. Props are...
   d) Mutable (can be changed)
   d) **Immutable (read-only)**
   d) Asynchronous
   d) Global

5. To use JavaScript expressions inside JSX, you must wrap them in:
   e) Parentheses ()
   e) Square brackets []
   e) **Curly braces {}**
   e) Quotation marks ""

Chapter 3

The Power of Hooks !

Hooks are functions that let you "hook into" React state and lifecycle features from function components. **Hooks aapko functional components mein superpowers dete hain, bina class likhe!**

3.0.1 useState: State ka Badshah

The most used hook to add state to components.

- **Kaise kaam karta hai?**: **‘useState‘ ek initial value leta hai aur ek array return karta hai. Array mein do cheezein hoti hain: pehla, current state value, aur doosra, ek function jisse hum uss value ko update karte hain.**

Example: A Simple Counter

```
import { useState } from 'react';

function Counter() {
  const [count, setCount] = useState(0); // [currentState,
   updaterFunction]

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>Click me</
   button>
      <button onClick={() => setCount(0)}>Reset</button>
    </div>
  );
}
```

3.0.2 useEffect: Side Effects ka Hero

Used for handling "side effects" like fetching data, subscriptions, or manually changing the DOM.

- **Kaise kaam karta hai?**: **‘useEffect‘ ek function leta hai jo har render ke baad chalta hai. Aap ek "dependency array" dekar control kar sakte ho ki yeh kab-kab chalega.**

- ‘[]‘ **(empty array)**: Sirf pehli baar render hone par chalega. (Perfect for API calls)
- ‘[count]‘ **(with state)**: Jab ‘count‘ state change hoga, tab chalega.
- **No array**: Har render par chalega. (Use carefully!)

Example: Fetching Data from an API

```
import { useState, useEffect } from 'react';

function GitHubUser() {
  const [user, setUser] = useState(null);

  useEffect(() => {
    fetch('https://api.github.com/users/ashu-kumar-13')
      .then(res => res.json())
      .then(data => setUser(data));
  }, []); // <-- Empty array means it runs only once!

  if (!user) {
    return <div>Loading...</div>;
  }
  return <div>Welcome, {user.name}!</div>;
}
```

Practice Zone
Master the ‘useState‘ and ‘useEffect‘ hooks with the official React documentation: useState Hook and useEffect Hook.

Your Challenge
Create a component that fetches a random dog image from the ‘https://dog.ceo/api/breeds/image/ran
API when it first mounts and displays it. Add a button that fetches a new image every time it’s clicked.

Quick MCQs

1. Which hook is used to manage state in a functional component?
   a) ‘useEffect‘
   a) ‘useContext‘
   a) **‘useState‘**
   a) ‘useReducer‘
2. To run an effect only once after the initial render, the dependency array should be:
   b) ‘[props]‘
   b) Omitted

b) ‘null‘

b) **‘[]‘**

3. What does ‘useState‘ return?

    c) An object

    c) A single value

    c) **An array with the state value and an updater function**

    c) A function

4. What is a "side effect" in React?

    d) An error in the code

    d) **Anything that affects something outside the component, like an API call**

    d) A component re-rendering

    d) A CSS animation

5. Can you use Hooks inside conditional statements (like ‘if‘)?

    e) Yes, always

    e) Only ‘useState‘

    e) **No, Hooks must be called at the top level of a component**

    e) Only in class components

Chapter 4

Handling User Interaction

4.1 Event Handling

React can listen to user events like clicks, keyboard input, etc.

- **Kaise kaam karta hai?**: **React events camelCase mein likhe jaate hain (jaise ‘onClick‘ not ‘onclick‘). Aap event handler function ko curly braces ‘‘ mein pass karte hain.**

Example:

```
function SignupButton() {
  const handleSignup = () => {
    alert('Welcome aboard!');
  };

  return (
    <button onClick={handleSignup}>
      Sign Me Up!
    </button>
  );
}
```

4.2 Lists and Keys

To render a list of items from an array, you use the ‘.map()‘ method.

- **Why Keys?**: **React ‘key‘ ka use karke identify karta hai ki list mein kaun sa item add hua, change hua, ya remove hua. Isse performance improve hoti hai. Har key unique honi chahiye apne siblings mein.**

Example:

```
const products = [
  { title: 'Cabbage', id: 1 },
  { title: 'Garlic', id: 2 },
  { title: 'Apple', id: 3 },
];
```

```

function ShoppingList() {
  const listItems = products.map(product =>
    <li key={product.id}>
      {product.title}
    </li>
  );

  return <ul>{listItems}</ul>;
}
```

Tip: Avoid using array indexes as keys if the list can be reordered, as it can lead to bugs. Always use a stable, unique ID.

Practice Zone
Read more about handling events and rendering lists in the official docs: Responding to Events and Rendering Lists.

Your Challenge
Create a 'TodoList' component. Define an array of todo items (e.g., '[id: 1, text: 'Learn React', ...]'). Render this array as an unordered list ('<ul>'). Make sure each list item ('<li>') has a unique key.

Quick MCQs

1. Which is the correct syntax for a click event in React?
   a) 'onclick=...'
   a) **'onClick=...'**
   a) 'on-click=...'
   a) 'click=...'
2. What is the primary purpose of using a 'key' prop when rendering lists?
   b) For styling purposes
   b) **To help React identify which items have changed, are added, or are removed**
   b) To use as an HTML 'id'
   b) It's an optional prop with no real effect
3. Which JavaScript array method is commonly used to create lists of elements in React?
   c) 'forEach()'
   c) 'filter()'

c) **‘map()‘**

c) ‘reduce()‘

4. A ‘key‘ should be:

   d) A random number

   d) The array index

   d) Globally unique in the application

   d) **Unique among its siblings**

5. How do you pass an argument to an event handler function?

   e) ‘onClick=myFunction(arg)‘

   e) **‘onClick=() => myFunction(arg)‘**

   e) ‘onClick=myFunction‘ and define the argument elsewhere

   e) You cannot pass arguments

Chapter 5

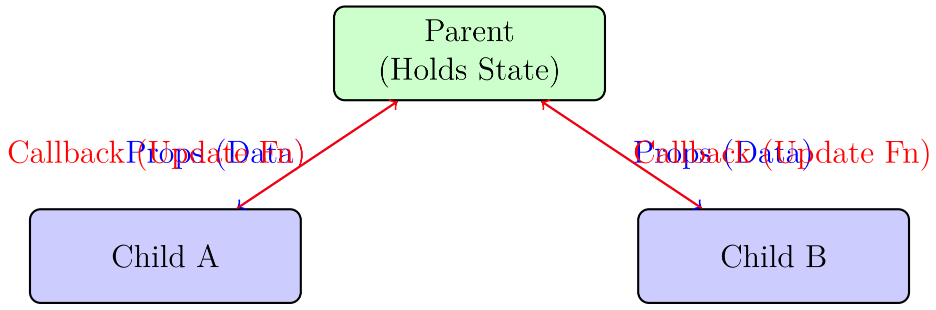
Advanced React Patterns

5.1 Lifting State Up

When multiple components need to share and reflect the same changing data, you should lift the shared state up to their closest common ancestor.

- **Kyun use hota hai?**: **Jab do bachchon (sibling components) ko ek hi khilona (state) share karna ho, toh woh khilona unke parent ke paas rakha jaata hai. Parent phir decide karta hai ki kaun sa bachcha kab khelega.**

Diagram: Lifting State Up



5.2 Context API: Global Data Share

Context provides a way to pass data through the component tree without having to pass props down manually at every level.

- **Analogy**: **Context API ek public announcement system jaisa hai. Ek ‘Provider‘ (announcer) data announce karta hai, aur koi bhi ‘Consumer‘ component, chahe kitna bhi deep nested ho, uss announcement ko sun sakta hai bina beech ke components ko disturb kiye.** This avoids "prop drilling".

Example:

```
// 1. Create a Context
const ThemeContext = React.createContext('light');

```

```
// 2. Provide the Context value at the top level
function App() {
  return (
    <ThemeContext.Provider value="dark">
      <Toolbar />
    </ThemeContext.Provider>
  );
}

// 3. Consume the Context value in any child
function ThemedButton() {
  const theme = useContext(ThemeContext); // Hook to read context
  return <button className={theme}>I am styled by the theme!</
    button>;
}
```

Practice Zone
Deep dive into these important patterns: Lifting State Up and Passing Data with Context.

Your Challenge
Create a simple application with a ‘theme‘ context. Make a ‘Switch‘ component somewhere deep in your app that can toggle the theme value (‘’light’‘ or ‘’dark’‘) held in the ‘App‘ component’s state. Any component that consumes the context should update its style accordingly.

Quick MCQs

1. "Lifting state up" means moving state to:
   a) A child component
   a) A global variable
   a) **The closest common ancestor component**
   a) A separate file
2. What problem does the Context API primarily solve?
   b) Slow rendering
   b) State management in large apps
   b) **Prop drilling**
   b) API data fetching
3. Which hook is used to consume a context value?
   c) ‘useState‘
   c) ‘useEffect‘

c) ‘useProvide‘

c) **‘useContext‘**

4. To provide a context value to the component tree, you wrap it in:

d) ‘<Context.Consumer>‘

d) ‘<Context.Value>‘

d) **‘<MyContext.Provider>‘**

d) ‘<Context.Injector>‘

5. Lifting state up is a good solution when:

e) Only one component needs the state.

e) **Two sibling components need to share the same state.**

e) You need to fetch data from an API.

e) You want to store data globally.

Chapter 6

Routing with React Router

React Router is the standard library for handling navigation in a React Single Page Application (SPA). **React Router se aap alag-alag pages bana sakte ho bina browser ko refresh kiye.**

6.1 Setup and Basic Usage

1. **Installation**:

```
npm install react-router-dom

```

2. **Configuration**: Wrap your ‘App‘ component with ‘BrowserRouter‘ in ‘main.jsx‘.
3. **Define Routes**: Use ‘Routes‘, ‘Route‘, and ‘Link‘ components to create your navigation structure.

Example:

```
// App.jsx
import { Routes, Route, Link } from 'react-router-dom';
import HomePage from './pages/HomePage';
import AboutPage from './pages/AboutPage';

function App() {
  return (
    <div>
      <nav>
        <Link to="/">Home</Link> | <Link to="/about">About</Link>
      </nav>
      <hr />
      <Routes>
        <Route path="/" element={<HomePage />} />
        <Route path="/about" element={<AboutPage />} />
      </Routes>
    </div>
  );
}
```

Practice Zone
Get started with the official React Router tutorial: React Router Tutorial.

Your Challenge
Create a small SPA with three pages: Home, Products, and Contact. Set up the routes and create a navigation bar with links to switch between these pages.

Quick MCQs

1. Which component is used to create navigation links in React Router?
   a) ‘<a>‘
   a) ‘<Route>‘
   a) **‘<Link>‘**
   a) ‘<Navigate>‘
2. To define the URL path and the component it should render, you use:
   b) ‘<Routes>‘
   b) ‘<Link>‘
   b) ‘<BrowserRouter>‘
   b) **‘<Route>‘**
3. Which component must wrap your entire application to enable routing?
   c) ‘<Routes>‘
   c) ‘<App>‘
   c) **‘<BrowserRouter>‘**
   c) ‘<Router>‘

Chapter 7

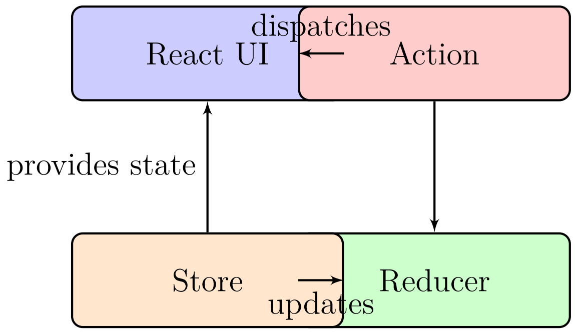
Advanced State Management: Redux

Redux is a predictable state container for JavaScript apps. It's used for managing complex, application-wide state. **Redux aapke app ka central brain hai, jahan saara important data (state) ek jagah store hota hai.**

7.1 Core Concepts of Redux

- **Store**: A single object that holds the entire state of your application.
- **Action**: An object that describes what happened (e.g., ‘ type: ’INCREMENT’ ‘).
- **Reducer**: A pure function that takes the current state and an action, and returns the new state.

Diagram: The Redux Data Flow



Modern Redux: Today, it's highly recommended to use **Redux Toolkit**, which simplifies Redux logic significantly.

Practice Zone
The official Redux Toolkit tutorial is the best place to start: Redux Toolkit Quick Start.

Your Challenge
Install Redux Toolkit and React-Redux. Create a simple counter application where the increment and decrement logic is handled by a Redux store.

Quick MCQs

1. In Redux, the entire application state is stored in a single object called the:
   a) Context
   a) **Store**
   a) Component
   a) Reducer
2. What is the only way to change the state in Redux?
   b) Calling a function directly
   b) Modifying the store object
   b) **Dispatching an action**
   b) Using ‘setState‘
3. A function that specifies how the app’s state changes in response to an action is called a:
   c) **Reducer**
   c) Dispatcher
   c) Store
   c) Middleware

Chapter 8

Performance, Testing Deployment

8.1 Performance Optimization

Keep your React apps fast with these techniques:

- **‘React.memo‘**: Prevents a component from re-rendering if its props haven’t changed.
- **‘useCallback‘ ‘useMemo‘**: Hooks to memoize functions and values, preventing unnecessary recalculations.
- **Code Splitting (‘React.lazy‘)**: Loads parts of your app only when they are needed.

8.2 Testing

Testing ensures your components work as expected. The standard toolset is **Jest** (for running tests) and **React Testing Library** (for rendering and interacting with components).

- **Philosophy**: **Test your components jaise ek user unhe use karega. Buttons dhoondo, click karo, aur dekho ki UI sahi se update hota hai ya nahi.**

8.3 Deployment

Time to show your app to the world!

1. **Build your app**: Run ‘npm run build‘. This creates a production-ready ‘dist‘ folder.
2. **Host it**: Deploy the contents of the ‘dist‘ folder to a static hosting service like **Netlify**, **Vercel**, or **GitHub Pages**.

Practice Zone
Learn about testing from the React Testing Library docs: Intro to RTL. For deploy-

ment, check out this guide for Vercel: Deploying React with Vercel.

Your Challenge
Take any of the apps you've built during the previous challenges, run the build command, and deploy it to Netlify or Vercel using their drag-and-drop interface. Share the live URL!

Quick MCQs

1. Which hook is used to memoize a function so it isn't recreated on every render?
   - a) 'useMemo'
   - a) 'React.memo'
   - a) **'useCallback'**
   - a) 'useEffect'
2. What is the purpose of code splitting?
   - b) To organize code into more files
   - b) **To improve initial load time by loading code on demand**
   - b) To make testing easier
   - b) To enable server-side rendering
3. Which command creates a production-ready build of your React app?
   - c) 'npm start'
   - c) 'npm test'
   - c) **'npm run build'**
   - c) 'npm deploy'

That's a wrap!

I hope yeh notes aapke liye helpful honge. React ek journey hai, not a destination. Keep learning, keep building, and most importantly, have fun! Happy Coding!