ARGUMENT OBJECT

- Arguments is a keyword which is used to fetch all the arguments Value inside the function without passing the parameter.
- It will return the argument object which is index data structure.
- We could not able to use argument keyword as identifier if the use script statement is there.

React

React is a popular JavaScript library developed by **Meta (formerly Facebook)** for building user interfaces, particularly for web applications. It allows developers to create reusable UI components that manage their own state, making the development process more efficient and scalable.

- **Component-Based Architecture**:
- React applications are built using small, reusable components that represent parts of the UI.
- Components can be nested and combined to create complex UIs.
- **Declarative Syntax**:
- Developers describe what the UI should look like, and React handles the updates to the DOM efficiently when the data changes.
- **Virtual DOM**:
- React uses a Virtual DOM (a lightweight copy of the real DOM) to optimize rendering. It updates only the parts of the DOM that have changed, rather than re-rendering the entire UI.
- **JSX (JavaScript XML)**:
- JSX is a syntax extension that allows developers to write HTML-like code within JavaScript, making the code more readable and expressive.
- **State and Props**:
- **State**: Represents dynamic data that changes over time.
- **Props**: Short for properties, they are used to pass data between components.
- **Unidirectional Data Flow**:
- Data flows in a single direction, making the application predictable and easier to debug.
- **Hooks**:
  1. Introduced in React 16.8, Hooks allow developers to use state and other React features in functional components without writing class components.
- **Ecosystem**:
  - React integrates well with other libraries and frameworks like Redux for state management or Next.js for server-side rendering.

Days = 3   27 nov 24

Single page web application (SPA)

- In single page web application, the reloading will not happen for every functionality we are performing.
- In single web application If we send the request to the server, we will get the response & again if we are sending any new request to the server, we will get the new response and the old response will get store it the browser cache.
- single page web application must faster in performs then MTA.

Multi page application (MTA)

- In multi-page web application reloading will happen for every functionality we are performing
- In multi-page web application, every time both old and new request will go the server and will get new response.
- multi-page web application is better SEO then single page web application
- To create multi-page we can use technology like j query (JavaScript library)
- To create single page web application we can use technology like (react, js library , framework , angular , veu)
- Mostly we can create e-commerce web-site any traditional website , like banking Educational website as a MPA
- We create social-media application or any streaming web application as spa EX Facebook, email

Library

- A library is collection of pre return that develops can use it preform some specific task
- The developer calls the library function method is needed
- The developer will control flow of application
- Library usually focus on particular functionalities or set of related functionalities
- Libraries are tool which you can use in your application
- Ex :- React , jQuery

Framework

- A Framework provides structured foundation or architecture for your application
- The framework controls the flow of program
- Developers can use component if need (Classes & functions).
- Framework is a set of libraries or package and it includes tool for multiple aspects of development
- Ex :- (Database, UI , Routing).
- Ex for framework :- Angular, node js , D jango

Days -4 **28 NOV 24**

Specifications of React

1.Component based architecture: -

React is used to create SPA(single page web application) architecture & application are build using reusable bits of code called component

Each component can manage own states and logic and we can use component multiple time.

2.Declarative in nature:-

Reacts allow developers to describe what the UI should like based on the application state.

3. Virtual DOM:-

- React application are faster in performance because its follows concept virtual DOM
- When the webpage is loaded both actual DOM and virtual DOM is created for webpage
- If we make any changes in the component UI first it will create new virtual DOM then it compares the both actual DOM and virtual DOM & makes the necessary update in actual DOM .

What is virtual DOM?

It is a light weight copy of real DOM which is used to optimize the update in DOM.

What is reconciliation?

- reconciliation is a process of comparing actual DOM and virtual DOM when the DOM updates or state of the component changes
- It will add only the changes which is made in virtual DOM to the actual DOM.

Diffing Algorithm: -

- when diffing two trees, react first compares the two root elements. The behavior is different depending on the types of the root elements.

NPM(Node Package Manager) & NPX(Node package execute)

NPM is the package manager for JavaScript that comes with node.js and it is used to install , uninstall & update the JavaScript packages or libraries required for development

- Manage dependencies in a project using package. JSON file

How to install packages locally and globally in NPM?

- npm install package-name –g (globally)
- npm install package-name (locally)
- For uninstall packages we can use the comment
- npm uninstall package-name –g (globally)
- npm uninstall package-name (locally)

NPX: -

- NPX is a tool bundle with NPM version 5.2 which is used to execute the packages directly without globally installing them
- To execute the package, we can use comment >npx package-name

Day = 5 29 november 2024

create-react-app package

its present in npm and its use to create new react application which involves basic project structure and necessary.

to install package we can use command = npm install create -react-app & -g

to create the react application we use the command create -react -app appName **(**name should not in capital letters**)**

app name should be in lower case it didn't accept upper case

npm start = Starts the development server.

npm run build = Bundles the app into static files for production

npm test = starts the test runner.

npm run eject = removes this tool and copies build dependencies configuration files and scripts into the app directory . if you do this . you can't goback!

after create the project to navigate app directory we can use the command CD appName.

Folder and file details of react app when we create using create-react-app package

a) node models it contents all the dependences and libraries install for the project.

We generally don't modified anything in node module folder and it handle by npm & yarn

b) public folder it contains static files and assists that will not process by webPack . and this static files we can use for our application . Inside this public folder we have one key file called index.html which in running on the browser when we start the folder

and it contain one element call div or rendering the react components.

c) src folder : it's a main directory where we can write react code (components , styles, etc) within the src mandatory we have to create one index.js file which is the entry point for the react app. The file renders the react app into the dom by targeting the div tag with id root in the index.html file.

Package.json file: it contains the metadata about project which include project name , description and version dependencies .

React package: It is used to create the root element for the react application so that we can make every component as a child of this root element using render()

It is present inside ReactDOM package

React.createElement():

This method is present within the react package and it is used to create new react element and it will take 3 arguments

1. element-type
2. attribute in the form of obj {},null
3. content " ",child

02 dec 2024 Day 6

JSX

- JSX stands for JavaScript xml, and it is used to write html code in react.
- JSX allows us to write the html elements in JavaScript and place them in the DOM without any create element and append child ()
- JSX converts html tags into react elements.

RULES OF JSX:

1) If we define multiple react elements mandatorily it should wrapped within the parent element i.e. one top level element should be present. Alternatively, we can use fragment also.
2) **Each and every tag must be closed.**
3) If multiple elements are there, it should be within the parenthesis {} .

CHANGES IN HTML ATTRIBUTES:

1) In react, class attribute should be named as className property because, class is the reserved word (key word) in JavaScript.
2) For attribute in label tag should named as html for property in react because for is also a keyword.

EXPRESSIONS IN JSX:

- With JSX, you can write the expressions within the {}.
- The expression can be a react variable or a property or any validation or any other JavaScript expression. JSX will execute the expression, and it will return the result.

FRAGMENTS:

- A fragment looks like and empty html tag (<> </>).
- Instead of taking one top level elements for JSX alternatively we can use fragments to wrap multiple lines this will prevent unnecessarily adding extra nodes to the DOM.

CONDITIONS IN JSX:

- React supports if statements but not inside JSX.
- We can use if statements outside JSX and we can take the result inside the JSX or we can use ternary expression inside the JSX.

COMPONENT IN REACT:

Components are the independent reuse able bits of code and returns JSX.

There are two types of components:

1) Class based component (stateful) (CBC)
2) Function based component (stateless)(FBC)

Class Based Components: (CBC) (stateful)

- These components are simple JavaScript classes made-up of multiple function that add functionality to the application.
- Every class components are **stateful** i.e. CBC we can define state with the help of state property (inbuilt property for every class based components).
- Every class component must inherit react. Components class and it should have render Method which returns JSX.

```
import React, { Component } from "react";
class App extends Component {
render() {
return (
<> </>

);
}
} export default App;
```

Function Based Components (FBC)

- These components are simple JavaScript functions (**arrow function , named function, function with expression**) that return JSX.
- Function based components are **stateless** i.e we don't have any inbuilt property called as state to define the data for that particular components.
- But after introduction of react hooks, we can able define the state in function based components also we can use usestate()hook

03-12-2024

State in react

- State holds the information or data about that particular component.
- We could not able to transfer the state from one component to another components.
- State are mutable (we can change).
- Whenever there is any changes in the state value the components will re-render.

How to define state in CBC?

- We can define state in two ways in CBC
- using **state property inside class** and **outside the constructor.** //1
- Inside the class using this.State inside the constructor(){}.// 2
- In CBC state value should be obj or null otherwise it will throws warning App.state : must be set to an object or null.
- Inside the constructor 1 statement should be the super calling statement which calls the parent class constructor.
- In CBC 1 constructor will execute after **that render()** will execute.

How to update the state value in CBC?

Every class component have one inbuilt method i.e setState() which is used to update state value in CBC and it will take one argument i.e value we are going to update (obj or null)

```
constructor() {
super();
this.state = { name: "ritick", };
}
```

Code for update

```
updateName = () => { this.setState({ name: "markus" }); };

<button onClick={this.updateName}>UPDATE</button>
```

4 dec 2024

React Hooks:

- React Hooks were introduced from the version react 16.8.
- Hooks allows function component to have access to state and other react features so that class components are generally not needed.
- Hooks allow us to use Lifecycle methods and state.

RULES OF HOOKS:

Hook types = useState, useEffect, useContext, useRef, useReducer.

There are three rules:
1) Hooks can only be called inside function components.
2) Hooks can only call at **the top-level of the component**.
3) Hooks cannot be conditional.

NOTE: Hooks will not work in react class based components.

We can also create custom hooks in react but it should start with use. Hooks are simple JavaScript functions which is used to perform some functionality in function based component.

How to define the state in Function Based Component?

• Function Based Components are stateless i.e. we don’t have any inbuilt property to define the state but we can **use useState** hook to define the state value.

• useState hook will take only one argument i.e. state value which can be any datatype and it will return one array with two elements. 1st is **Statevalue** and 2nd is **updater function.**

• This updater function is used to update the state value and it will take one argument i.e. the value we are going to update.

Events in React:

• Events in React are triggered by user action such as clicking the button or handle by eventhandlers.

• We have to pass the eventhandler property such as onClick, onSubmit in the opening tag of the react element.

• This property will take one function as the value and this function will execute when the use triggers the event.

Date = 05 / 12 /2024

Handling events react:

- We can handle the events by passing handler property in the opening tag of the react elements and every event handler property.
- Will take one function as a value and that function will execute if the user triggers the event.
- We can pass arrow function or named function as a value .
- But incase of class-based component when we are passing named function or anonymous function this keyword refers to undefined and we have to explicitly set the value for this key word as the current component (same component).
- We can set the value for this keyword inside the constructor using bind method.
- Example

This.handleMouseOver =this.handleMouseOver.bind(this);

Passing function as an argument for the updater Function In Class-based Component Or function based component:

- To update the state value, we can also pass a function as an argument, and that function will take one parameter that holds the previous value.
- In the return statement, we have to pass the value that we are going to update.
- We can use this for both class-based components or function-based components.

06 DEC 2024

Props:

- Properties
- Transfer the data from one component to another
- Immutable
- Data flow from parent to Child.

- **Props are properties which is use to transfer the data from one component to another.**
- Props are immutable (we can't change its value).
- Props can pass any types of data string, number, array etc.
- Props follow a unidirectional data flow i.e. from parent to child

Passing data through props:
The syntax for passing data from 1 com to another is same in cbc & fbc

```
<Child name={"abc"}
        id={123}
        skills={["js","java","pyhton"]}
        demo={demo}
        />
```

When we are passing the staring data no need to write within the expression directly we can pass within " ",' '

Receiving the data in child component:

Receiving data in child component

-> In CBC to receive the props we should us **this.props** property and it will return each and every data in a form of object

-> In FBC we can receive the props by passing parameter this also return the data in form of object.

10-12-2024

How to set default props:

In CBC we can set default Prop using three way.

1. Using static property default Props inside the component.
2. Using OR (||) operator
3. Using defaultprops property outside the component.
   syntax : component_name.defaultProps={prop: "default_value"}

In FBC we can use default props in 2 ways

1. Using OR(||) operator
2. Using defaultprops property outside the component
   syntax :component_name.defaultProps={prop: "default_value"}

How to set the datatypes of props that we are received in child component

Step1: install the package i.e props-types
Step2: import {props Types} from ""
Step3: we can use the property prop-type outside the component and we can set the datatypes for props
Syntax : comp_name.proTypes ={
Prop: proptypes.string;
Prop: proptypes.number;
Prop: proptypes.func;
Note: React Follows unidirectional dataflow i.e through props we can pass the data from parent to child Component. Through props we couldn't able to transfer data child to parent comp. We can use call back function in order to send data from Child to parent comp.

Refs in react
In react refs is an attribute or property that provides a way to directly access or interact with DOM element or react component.
How to create the ref:

- In CBC we should use the method React.createRef() & It will return ref object with key current &value as null.
- To create the ref, we can create the ref object inside the constructor in CBC.
- If FBC to create a ref object we can use HOOK useRef() & it will return obj with key current and value undefined.
- To pass the reference for the react else we should use ref{} property in both CBC & FBC.

NOTE: Don't over use ref only use the ref only needed otherwise it will affect the performance .
Common use case for ref:
- Managing focus, text Selection.
- Controlling Media Players.
- Triggering animation.
- Integrating with 3rd party Libraries that requires direct dom manipulation.

Conditional rendering on React:

We can render a component based on same component It have been done in 4 ways:
1. If else
2. Switch case
3. Ternary operator
4. && operator

If else statement we couldn't able to pass inside JSx
But we can pass ternary operator and && operator

We can also use nested component in order to perform conditional rendering.

Date = 12-12-24

Controlled component & Controlled component.

- Controlled component
  In cc the form data is handled by react state
  The value of form element like input text area id are set by component state any changes to form element that we can update using eventhandlers like onChange.

- UnControlled component:
  In uncontrolled component the form data is handled by DOM state.
  The ref attribute is used to access the DOM node to get or set value.
  React does Not manages state of input instead of browser does.

Date: 24 December 2024

Component life cycle

In react Component life cycle refers to sequence of events that happens during components existence.

There are 3 phases of life cycle	1. Mounting,	2. Updating ,	3. unMounting

1. **Mounting** → Component is created and it is placed in DOM.

I. Constructor → initialize the state, bind the event handlers.
II. static getderivedstatefromprops(props, state)→ return object or null
III. Render()
IV. ComponentDidMount ()
V. Side Effects in reacts.

2. **Updating** → Changes in State / Send any props.

I. Static get derived State from props
II. Should component update()
III. Render()
IV. getSnapShotBeforeUpdate()-> props , state
V. component did update()
VI. component DidUpdate()→ props,state, snapshot value.

3. **Unmounting** → If Component is removed from the Dom.

Mounting phase methods

Constructor

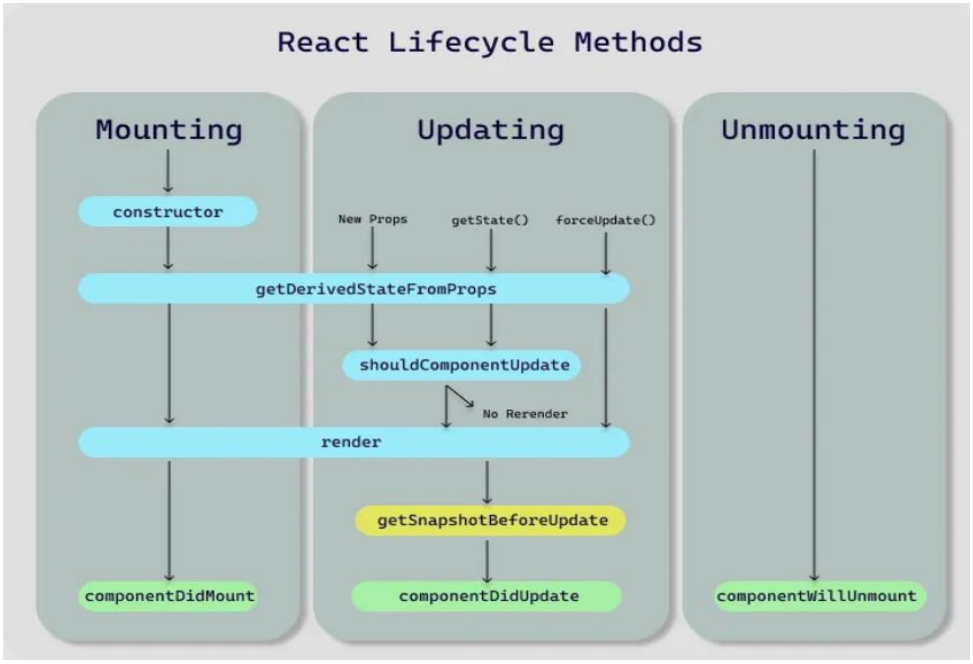
<u>It is the best place to initialize the state and also used to bind the event handler</u>

It is not suitable for performing side effect.

Static getDerivedFromProps(props, state)

This method is used the update the state value that depends on props
The value we have going to update for the state should pass in return statement otherwise null
It will take 2 parameter 1st is props 2nd is state
It is also not suitable to perform side effect
Render()
Must used method in CBC and returns JSX
It also not suitable to perform side effect
ComponentDidMount ()
It is used to perform side effect like data fetching ,manually changing the dom and setting up subscriptions
Side Effects in reacts
In React, “side effects” refer to any operations or behaviors that occur in a component after rendering, and that don't directly impact the current component render cycle. These side effects can include tasks such as data fetching, subscriptions, manually changing the DOM, or other interactions with the outside world.

25 December 2024



Updating phases method

1. Static getDerivedstateFromProps
2. should component Update

→this method define weather component should re-render or not.
→ it will return Boolean value , if return value if true component will re-render.
If false then component will not re-render .

3. render()

4. getsnapshotbeforeupdate()

→this method is used to store the previous state and previous prop before getting updated.
→The value which we are going to store should be passed in return statement.
→this method should be used along with componentdidupdate() otherwise it will throws warning.
→This snapshotvalue we can fetch inside componentdidupdate by passing only in third parameter

5. componentdidupdate()

→This method is envoke immediately after component has been updated.
→It is commonly used for performing side effects like:- data fetching, manually updating DOM

It will three parameter

1. previous props
2. previous state
3. snapshot value

unmounting phase method

componentwillunmount()
this method is call just before the component is unmounted or destroyed.
→It is used for cleanup task such as cancelling network request, removing event listeners.

Note: If we have parent & child component & Whenever child component is in unmounting phase before execution of componentwillunmount method in child component.

getSnapshot Before Update method in parent component is getting executed.

This is because to store the previous prop & previous state before child component is removed from DOM tree.