Component Life Cycle:

Each Component goes through life cycle. There are three phases such mounting, updating, unmounting

1. Mounting

Mounting Means putting element into DOM

There are four build in method get called during mounting phase

Constructor() :

* Constructor method is called before anything else. It is intialise state and other initial value

getDerivedStateFromProps () :

* Method() is called right before rendering the elements in the DOM.
* It takes state as argument and return an object with changes to the state
* Method allows you modify state value from prop value

Render( ) :

* render() method is required and this method Output HTML to DOM .

componentDidMount() :

* Method() is called after the component is rendered
* This is where you run statement that require that component already places in the DOM

construction => to set state value and other value

getDerivedStateFromProps => set state value from initisl props

render => show html output on user intercae

componentDidMount => perform certain operation after redering take place

getDerivedStateFromProps => same as mounting

ShouldComponentUpdate => return boolean value that determine react should continue rendering or not

reder => same as mouting

getSnapshotBeforeUpdate => use to maintain perv state and props value

componentDidUpdate => update component after reder

componentWillUnmount = component is removed from DOM

Life Cycle Method in Functional Component.

With help of useEffect() hook we can achieve life cycle method in functional components

The useeffcts() hooks has two arguments – a callback function and dependency array. The callback function is mandatory while dependency array is optional

1) The componentDidMount() lifecycle method executes when component is mounted. Means when component is added to DOM tree, this method is excutes

The empty dependency array means that the hook will execute only once and will be during the component mounting

When first time rendering happen it will treat as componentDidMount() afterward for each rendering it treat as componentDidUpdate()

2) The componentDidUpdate() lifecycle method will execute when component is updated.

Now I created one state variable with help of useState hook having name count and initial value 0 and now will increment that variable when button click attach one onclick event listener will call callback function which increment value. So when value of state variable update component will re-render and useffect() excutes.

3)The componentWillUnmount() lifecycle method excutes when the component is unmounted or remove from the DOM tree.

If the function is remove from the useEffect hook. It will be excuted when the component is unmounted

Thus behaving like componentWillUnmount

Hooks Documentation :

1. What is Hooks : -

* Hooks are added in react 16.8 version
* They are build in features allows to use state and life cycle method in functional component. Only use in functional component
* Using Hooks, we can extract the stateful logic from a component so it can be tested independently and reused.
* In functional component hooks are always define at top level. They should not call inside nested function, loop or condition

1. What is use of useState() Hook : -

* useState() hook allow to track of state in functional component
* the term state refer to data or properties of component in react
* first we have define usestate() with initial value, which returns variable with state value and another function to update the current state of the variable

1. What is use of useEffect() Hook :-

* useEffect is used to manage side effects of functional components
* side effects refer to fetching data, DOM manipulation, timer function
* accept two argument a callback function and dependencies
* the callback function contain side effects, while dependencies are optional

1. What is use of useRef() Hook :-

* It allows you to persist values between render

1. What is use of useCallback() Hook :-

* It is used to prevent declare function within body of function to avoid to recreate function on every render
* Use in performance improvement

1. What ia use of useMemo() Hook : -

* It is use to remembering past value that is already been captured
* Use in performance improvement

1. useContext Hook :-

Context API allows to access data at different level without passing data to several components

First we need to define context using createContext() then pass data through custom component provider method with value attribute

Then export that data to respective component and then import that data with help of usecontext() hook with take context object/name as argument and return value of that context

What is React Context API

Context API allow to access data at different level of the component tree without passing prop to every level

Main aim of context api to solve problem of prop drilling

1) First we have to create context object and define data which we want to store.

2) Use context provider through which we pass data

3) Use context consumer who gets that data

What Is Ref

Ref is shortend used for reference it is an attribute to store reference of particular node or react elements react property with help we manipulate DOM.

Refs are used to access DOM elements. And perform DOM manipulations

Ref is created by using useRef hook

Use Cases of ref :

* Managing focus, text selection or media playback
* Triggering Animation

What is useRef()

It is used to stored a mutable values that does not cause re-render when updated.

It is also use to access DOM element directly

What is useReducer

useReducer() is similar to useState() use for state management

It accept two arguments reducer and initial value and return array with two elements state and dispatch(update function)

Reducer function accept two argument state and action

Dispatch() is used to perform operations dispatch trigger action method of reducer function where we play with state

In dispatch method we have to pass one object in that type property is mandatory

useReducer is suitable for managing complex state.

It allows you to define a separate action object and switch statement to handle different state updates.`

What is useMemo

useMemo() is same as useffect() hook but useeffect() does not return value useMemo() return value and will store in variable

useMemo() function take two arguments a function compute that computes the result and the dependency array

During the initial redering call compute function, memorize caculation result for next time and return it to its component

If the dependencies don't change during the next renderings, then useMemo() doesn't invoke compute, but returns the memoized value

But if the dependencies change during re-rendering, then useMemo() invokes compute, memoizes the new value, and returns it.

The useMemo hook is used to improve performance in our React application.

The useMemo is a hook used in the functional component of react that returns a memoized value.

memoization is a concept used in general when we don’t need to recompute the function with a given argument for the next time as it returns the cached result.

A memoized function remembers the results of output for a given set of inputs

if there is a function to add two numbers, and we give the parameter as 1 and 2 for the first time the function will add these two numbers and return 3, but if the same inputs come again then we will return the cached value i.e 3 and not compute with the add function again

Semantic HTML

It clearly define structure and purpose of content

What is useCallback

useCallback() function memorize function

useCallback() hook is used to improve performance in our React application.

closure

callback

promise

pure component

HOC

life cycle in class component

how you will implement in functional component

symantic html

psudeo element

position in css

garbage collector

object prototype

features of react

redux

testing library

diff between normal/ arrow function

what is classe