# Assignment - Ch-04

SAGAR
Page No.
Date

① Is JSX mandatory for React?

Ans- No, JSX is not mandatory for React we can also use React.CreateElement but that is hard to read, write & understand.

② Is ES6 mandatory for React.

Ans No, Not at all, Nothing is mandatory for React; we can also vanilla JS and ES6. there is no issuce in using older version of JS.

③ How can I write comments in JSX

Ans we can write @ by using {
- (i) Single line //
(ii) multiline /* */
inside { }
same like JS.

```
Ex- {
    // same like JS
    /* yes, it is
       same like JS */
}
```

④ What is <React.Fragment> </React.Fragment> and <> </>.

- In react we return multiple element under single parent but always we can't prefer to use parent like ex- div so, we use React.Fragment — it help us to return element

SAGAR
Page No.
Date

Q Component without wrapping inside parent element.

React fragment to use every time it is hard to write so, shorthand for React fragment <> </>.

Ans What is Virtual Dom?

In general, Virtual Dom is representation of Dom. ~~in our case~~ we need it for reconcilliation as it uses diff algorithm to find out the difference b/w tree and re-renders only the difference part.

In detail :-

Virtual Dom is data structure of HTML Dom in memory. it is tree of objects that corresponf to the elements in HTML document. Each object in Virtual Dom is called Virtual Node.

when user interact with web application such Clicking or typing, the state of Abplication change This change triggers a process called reconciliation. where Virtual Dom compare the current State of application with previous state and calculate difference b/w them. This process is also know as diffing algorithm

The diffing algorithm works by comparing each virtual Node in old virtual Dom with its corresponding in new virtual Dom. if a virtual Node has changed, the algorithm updates its properties and re-renders it. if a virtual dom has been added or removed the algorith creates or deletes in the HTML DOM.

"" to optimize diffing algorithm, virtual dom uses technique is called batching. Batching groups multiple changes to state of application & performs the diffing algorithm only once, instead of after each change. This technique reduce the no. of updates to Dom & improve performance''.

Q4 what is Reconciliation in React?

& Reconciliation is process of updating the user interface to reflect changes in underlying data model, when data or state changes in a React component, React decide how to efficiently update Dom to reflect those changes To do this, React perform a process called Reconciliation.

- During reconcilliation process, React create a new virtual Dom tree and compare it with previous virtual Dom tree. it then identifies the difference b/w the two trees

and updates only the necessary part of the Dom to reflect the change.

In detail

When comparing two React Dom elements of the same type, React looks at the attributes of both, keeps the same underlying Dom node, and only updates the changed attributes.

- Before updating component, React will call the 'UNSAFE_componentWillRecieveProps()' 'UNSAFE_componentWillUpdate() and componentDidUpdate()' methods on the instance. These methods allow Component to perform any necessary action before & after the update such as updating the state or making network requests.

---

## What is React Fiber?

- React Fiber is internal implementation of React Core algorithm for rendering components and updating user interface in response to changes in state or props. It was introduced in React 16.

Fiber is a reimplementation of the React reconciliation algorithm that

enable better control over rendering process. it break down rendering pipeline into smaller units of work called fibers that can be prioritized, scheduled, and interrupted if necessary, allowing React to perform work in smaller chunks and avoid long pause that can lead to janky user interface.

main benefits of React Fiber is:-

- improved performance:- Fibre allow React to prioritize and schedule rendering work more efficiently, resulting in faster rendering and improved frame rate
- more controll over rendering:- Fiber enable developers to interrupt and resume rendering work which can be useful for implementing features like progressive rendering, lazy loading, and user-driven interaction.
- improved developer experience: Fiber provide a more predictable and consistent API for working with React rendering engine making it easier to reason about & debug React application.

SAGAR
Page No.
Date

2 Why we need keys in React ? when do we need keys in React ?

Ans Keys help React identify which items have changed, are added or are removed.

- key must only be unique among siblings - keys used within array should be unique among their Siblings ; however they don't need to be globally unique. we can use same keys when we produce two different arrays :- ex -

```
Function Blog (props) {
    Const sidebar = (
        <ul>
        {props.posts.map((post) =>
        <li key={post.id}>
            {post.title}
        </li>
        )}
        </ul> );
    Const content = props.posts.map((post) =>
        <div key={post.id}>
        <h3> {post.title} </h3>
        <p> {post.body} </p> </div> );
```

- Rendering multiple Components :- when rendering multiple Components of the same type, React needs to know which Component is which. By providing a unique key for each Component, React can efficiently update components
- Rendering dynamic lists or Re-ordering or filtering a list -

SAGAR
Page No.
Date

Q) Can we use index as key in React?

Ans yes, it is possible to use index as keys in React but it is generally not recommended and should only use as last resort in certain situations.

1) Performance issue! - using index as a key can lead to performance issue in large list or when dealing with complex component. This is because when a new item is added to list or an item is deleted or moved, React will have to re-render the entire list even if single item has changed.

2) Incorrect state :- as React may not correctly update the state of child component if an item in the list is deleted or moved. This can leads to bugs or unexpected behaviours, particularly if child component depend on state of list to function correctly.

3) Unpredictable behaviour :- when using index as key, the order of list can affect the behaviour of component, particularly if list is dynamic and can be reordered or filtered.

index as key can lead

<ul>
<li key="0"> 1 </li>
<li key="1"> 2 </li>
<li key="2"> 3 </li>
</ul>

Add new start

<ul>
<li key="0"> 1 </li>
<li key="1"> 2 </li>
<li key="2"> 3 </li>
<li key="3"> 4 </li>
</ul>

SAGAR
Page No.
Date

Index as key Should be used only
(1) The item in list do not have unique id.
(2) The list is static list & will not change.
(3) The list never be recorded or filtered.

Q) What is Props in React ? ways to

Props is shorthand for properties and refer to mechanism for passing data. it is normal parameter like in javascript parameter function

it is convientional to use Props but not mandatory

```
Const List = (Props) => {
  return (
    <div> {Props} </div>
  )
}
```

1 Inline props for ex:-

```
<myComponent foo = "bar" />
```

foo prop is passed to 'myComponent' with value 'bar'

2 Via Variable : 'Props' can be also passed via variables when component is rendered for ex

```
Const myProps = {
    foo: 'bar'};
<myComponent {...myProps} />
```

SAGAR
Page No.
Date

Here 'foo' prop is defined in 'myprops' variable & then spread into 'myComponent' & combined using spread syntax

```
function myComponent (Props) {
    return <div> {props.foo} </div>;
}
```

3- Default Props - we can define default values for Props using 'defaultProps' static property of component.

```
function MyComponent (Props) {
    return <div> {Props.foo} {props.bar} </div>;
}

MyComponent.defaultProps = {
    foo: "default value for foo",
    bar: "default value for bar",
};
```

here, default values are defined for the 'foo' and 'bar' props using defaultProps static property of Component.

1mb

what is a Config Driven UI