## Is JSX mandatory fot react?

NO, JSX is just syntactic sugar for writing the react code that is more readable and maintainable. Without JSX we need to use React.createElement that is having littile big syntax than JSX.

ex by using JSX--

```

let pTag=<p>P tag with JSX</p>

```

by without using JSX--

```

let pTag=React.createElement("p",{},"P tag without jsx")

```

Sometimes use of JSX avoided when we are going to create a small application and we want to skip compile part.

## Is ES6 mandatory fot react?

NO, ES6 is a standard implemented by JavaScript that makes the code more readable and accessible.There are many features that was introduced in ES6 that helps the developer to write complex code in a easy way. Some of the ES6 features that are widely used in React are destructring,classes,promises and default props as the property of class component.

## Comment in JSX

### Single Line Comment-

A single line can be commented by using double forward slash //

ex-

```

//let d="Single line comment"

```

### Multiline Line Comment-

We need to use JavaScript comments inside of Curly braces like {/\*comment here\*/}. It is a regular /\* Block Comments \*/, but need to be wrapped in curly braces.

ex-

```

{/\*

let d=" line1 comment"

let e=" line2 comment"

\*/}

```

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

These both things are same <></> is just a shorthand for <React.Fragment></React.Fragment> and these are known as react fregments.

We can return only one element from a react component but it can have multiple children but sometimes we need to return more than one element but we can't wrap them in a div in this situation we use React Fregment to wrap them all in one element and this react fragment is not visible in DOM tree it is the same as if the elements were not grouped.

ex-

```

function Post() {

return (

<>

<PostTitle />

<PostBody />

</>

);

}

```

## Virtual DOM

When ever a web page is got rendered react creates a copy of actual dom and stores that in memory for future updation of actual dom optimization. It is stored in memory as a JavaScript object and it has all the important properties similar to actual that are required to compare the virtual dom tree to actual dom tree and identify which part of dom is actually changed .

## Reconciliation

when ever there is some update to show on screen we compare the virtual dom to actual dom to identify which part of dom needs to be updated or changed . The algorithm that React uses to compare or take diff of actual and virtual dom is known as reconciliation process.

Before react used to use stack alogorithm but it uses more efficient and optimized algoritm called Fiber.

## React Fiber

React Fiber is new reconciliation algorithm it is based on two assumption

1-Different component types are going to create different trees.React will not diff them and directly replace them with one another.

2-List elemnts are compared using key attributes.So key values should be stable, predictable and unique.

### Goals acheived by fiber

1-Pause work and come back to it later.

2-Assign priority to different types of work.

3- Resuse previously completed work

4- abort work if it is no longer needed.

## Need of Keys in React

As react Fiber compares list elements by the keys in old virtual dom and new virtual dom. While list items compared fiber first compares key if key is same than it is assumed that that li element is not changed. If we do not provide key to li elemnets than react fiber will compare each and every property of all li tag that is not a optimized behaviour.

ex-

```

<li key={0}>1</li>

<li key={1}>2</li>

<li key={2}>3</li>

```

## Can we use indexes as Keys

Yes we can use indexes as keys if our list or array not going to be manipulated (adding or deletion at end or start) and sorted otherwise it can result to a unstable behaviour.

## Props--

Props are the way by which component communicate with each other.As we know component are javacript functions so props are only arguments that we can pass to functional component.

whatever props we want to pass to a component we give that as attributes and these all get wrapped into one objects get passed to the functional component.

ex-

```

function compo(props){

return <p>props.text</p>

}

<compo text={passed props}/>

```