REACT JS (JAVA SCRIPRT)

INTRODUCTION OF REACT JS

Q)1.What is React JS ?

Q)2.History if React JS?

Q)3.What is Dom tree?

Q)4.diffrence b/w React and virual Dom?

Q)5.how dose react works?

Q)6.What is compontes?

Q)7.What is JSX?

Q)8.Diffrence b/w react js and react native?

Q)9.Diffrence b/w react js and node js?

Q)10. why we install node js and Npm?

# What is React?

React, sometimes referred to as a frontend JavaScript framework, is a JavaScript library created by Facebook.

React is a tool for building UI components.

# React.JS History

Current version of React.JS is V17.0.2 (August 2021).

Initial Release to the Public (V0.3.0) was in July 2013.

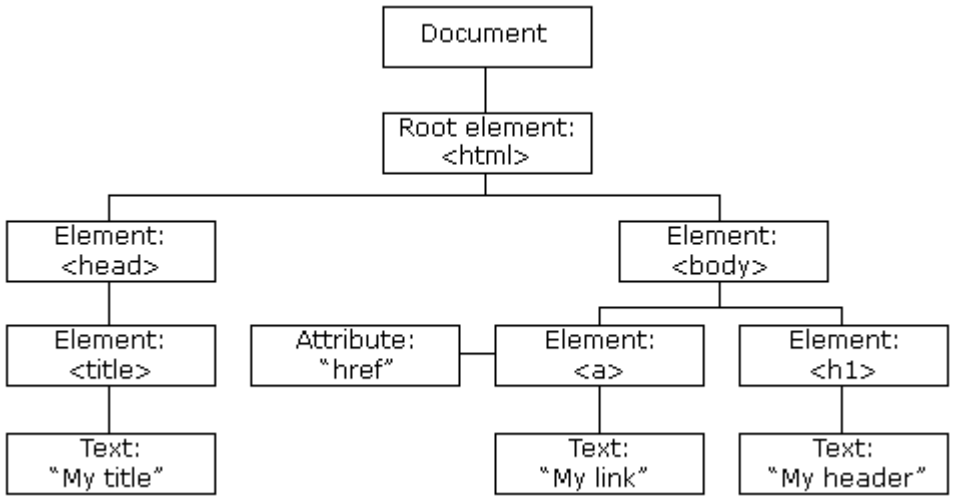
React.JS was first used in 2011 for Facebook's Newsfeed feature.

Facebook Software Engineer, Jordan Walke, created it.

Current version of `create-react-app` is v4.0.3 (August 2021).

`create-react-app` includes built tools such as webpack, Babel, and ESLint.

**DOM:** DOM stands for 'Document Object Model'. In simple terms, it is a structured representation of the HTML elements that are present in a webpage or web-app. DOM represents the entire UI of your application. The DOM is represented as a tree data structure.




## What is the Virtual DOM?

The virtual DOM (VDOM) is a programming concept where an ideal, or "virtual", representation of a UI is kept in memory and synced with the "real" DOM by a library such as ReactDOM. This process is called reconciliation.

# How does React Work?

React creates a VIRTUAL DOM in memory.

Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM.

React only changes what needs to be changed!

React finds out what changes have been made, and changes **only** what needs to be changed.

You will learn the various aspects of how React does this in the rest of this tutorial.

# React Components:

Components are like functions that return HTML elements.

# React Components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML.

Components come in two types, Class components and Function components, in this tutorial we will concentrate on Function components.

# Create Your First Component

When creating a React component, the component's name *MUST* start with an upper case letter.

## Class Component

A class component must include the `extends React.Component` statement. This statement creates an inheritance to React.Component, and gives your component access to React.Component's functions.

The component also requires a `render()` method, this method returns HTML.

### Example

Create a Class component called `Car`

```
class Car extends React.Component {
  render() {
    return <h2>Hi, I am a Car!</h2>;
  }
}
```

## Function Component

Here is the same example as above, but created using a Function component instead.

A Function component also returns HTML, and behaves much the same way as a Class component, but Function components can be written using much less code, are easier to understand, and will be preferred in this tutorial.

### Example

Create a Function component called `Car`

```
function Car() {
  return <h2>Hi, I am a Car!</h2>;
}
```

# Rendering a Component

Now your React application has a component called Car, which returns an `<h2>` element.

To use this component in your application, use similar syntax as normal HTML: `<Car />`

### Example

Display the `Car` component in the "root" element:

```
ReactDOM.render(<Car />, document.getElementById('root'));
```

# Components in Components

We can refer to components inside other components:

### Example

Use the Car component inside the Garage component:

```
function Car() {
```

```
  return <h2>I am a Car!</h2>;
}

function Garage() {
  return (
    <>
      <h1>Who lives in my Garage?</h1>
      <Car />
    </>
  );
}

ReactDOM.render(<Garage />, document.getElementById('root'));
```

# What is JSX?

JSX stands for JavaScript XML.

JSX allows us to write HTML in React.

JSX makes it easier to write and add HTML in React.

# Coding JSX

JSX allows us to write HTML elements in JavaScript and place them in the DOM without any `createElement()` and/or `appendChild()` methods.

JSX converts HTML tags into react elements.

## Example 1

JSX:

```
const myelement = <h1>I Love JSX!</h1>;

ReactDOM.render(myelement, document.getElementById('root'));
```

## Example 2

Without JSX:

```
const myelement = React.createElement('h1', {}, 'I do not use JSX!');


ReactDOM.render(myelement, document.getElementById('root'));
```

| Node.js | React.js |
| --- | --- |
| Node.js used as a back-end framework | React is used for developing user interfaces. |
| It supports the Model–view–controller (MVC) framework. | Does not support the Model–view–controller (MVC) framework. |
| It runs on chrome’s v8 engine and uses an event-driven, non-blocking I/O model, which is written in C++. | It uses Node.js to compile and optimize the JavaScript code and easy to create UI Test cases. |
| Node.js handles requests and authentication from the browser, make database calls, etc. | It makes API calls and processes in-browser data. |
| Here the Real-time data streaming is handled easily. | In React complex architecture makes it hard to keep track of the traditional approach. |
| Framework for JavaScript execution having the largest ecosystem of open source libraries. | Facebook-backed Open Source JS library. |
| The language used whichonly JavaScript. | The language used is JSX and JavaScript. |

| Node.js | React.js |
| --- | --- |
| There is no DOM (Document Object Model) concept that is Used. | Here the Virtual DOM (Document Object Model) is Used that makes it faster. |