1) What is React Js?

->

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

React is a tool for building UI components.

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.

To get an overview of what React is, you can write React code directly in HTML.

But in order to use React in production, you need npm and [Node.js](https://nodejs.org/) installed.

2) What is NPM in React Js?

->

NPM is short for node package manager, an online directory that contains the various already registered open-source packages. NPM modules consume the various functions as a third-party package when installed into an app using the NPM command npm install

Third-party NPM modules are the unit of logic for a specific functionality or a full-fledged library, such as reactstrap, material-ui, redux-form, and so on. In this guide, you will learn how to load NPM modules in your app using the NPM command or the GitHub source code.

Load the NPM module using the CLI

Any NPM modules can be easily consumed in your app using the CLI command followed by the module name. Below is the syntax used to install the NPM module.

Syntax:

Npm install <module\_name>

installing any module using the above CLI command. Let's install reactstrap for an example.

Npm install reactstrap

When you execute the above command, the related module folder will get added to the directory node\_modules in your current app, and you will be able to use the components or functionality of the module anywhere in your app.

3) What is Role of Node Js in react Js?

->

Node is the most popular platform for hosting and running web servers for React apps. After all, using NPM (Node Package Manager), you can install any package with NPM command-line interface CLI. Moreover, Node utilizes Webpack and other modules to turn a React application into an easily compilable single file.

## Why Use the Combination of Node.js and React for Web App Development?

There are specific reasons to use the deadly combination of Node.js and React for web app development.

### 1. Scalability

The combination of Node and React allows developers to build dynamic, large, data-driven web apps that are responsive across multiple devices. It would help if you had scalability while working on large projects and maintaining your website’s performance.

### 2. MERN Stack

MERN Stack means MongoDB, Express, React, and Node.js. And no better combo than these frameworks can provide a unique dimension to a website. You can use Node.js with MERN Stack alongside React.js. Hence, you can use the Node and React combination for web app development.

### 3. JSON APIs

Creating JSON (JavaScript Object Notation) APIs for web development is competent due to the high code reusability and access to immediate code sharing in React.js. And Node.js can effectively allow this.

### 4. Real-Time Data

If your business app handles real-time data management or aims to build a data streaming app, you should use Node.js as your app requires an ongoing server connection.

### 5. Fast Development

When using the combination of React and Node for [web app development](https://www.mindinventory.com/web-development.php), you can receive a high ROI and save your money and time. After all, these technologies work excellently to offer an effective platform for building rapid functioning and easy-to-maintain websites.

### 6. SPAs (Single Page Applications)

Does your business require a single-page app with asynchronous data loading? Then, you must [select React](https://www.mindinventory.com/blog/top-reasons-to-choose-reactjs-for-your-project/) with Node back-end as it allows building a lightweight back-end model through callback functions.

### 7. Single Language for Front-end and Back-end

With the combination of Node and React, developers don’t require learning complex back-end languages like Python or Ruby. They can use Node for server-side development and React for front-end code building without switching between frameworks and programming languages. And it saves resources, money, and time.

### 8. High Server Load

The combination of Node.js and React can balance the high server requests and load when developers work on web app development.

### 9. Organized Process

The deadly combination of React and Node create an organized web development process. After all, these technologies are scalable, effective, and fast. When working together, they can help you build high-functioning websites.

### 10. Increased Use of JavaScript

Node with React enables the use of the total strength of JavaScript for building front-end and back-end codes. It gives more freedom and convenience when building websites or web apps, as you can use one language for every aspect.

4) What is CLI command In React Js?

->

the basic command available in Create React App command line application in this chapter.

Creating a new application

*Create React App provides* multiple ways to create React application.

Using *npx* script.

npx create-react-app <react-app-name>

npx create-react-app hello-react-app

Using *npm* package manager.

npm init react-app <react-app-name>

npm init react-app hello-react-app

## Selecting a template

*Create React App* creates React application using default template. Template refers the initial code with certain build-in functionality. There are hundreds of template with many advanced features are available in npm package server. *Create React App* allows the users to select the template through *–template* command line switch.

create-react-app my-app --template typescript

## Installing a dependency

React dependency package can be installed using normal *npm* or *yarn* package command as React uses the project structure recommended by *npm* and *yarn*.

Using *npm* package manager.

npm install --save react-router-dom

## Running the application

React application can be started using *npm* or *yarn* command depending on the package manager used in the project.

Using *npm* package manager.

npm start

5) What is Components in React Js?

->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.

In older React code bases, you may find Class components primarily used. It is now suggested to use Function components along with Hooks, which were added in React 16.8. There is an optional section on Class components for your reference.

### **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:

Class car extends React.Component{

Render(){

Return(

<>

</>)}

### **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

Function car(){

return <h2>Hi, I am a car!</h2>

}

6) What is Header and Content Components in React Js?

->

Header:

The Card can be created with header title, sub title and images. For adding header you need to create div element with the class e-card-header added.

Card provides below elements and corresponding class definitions to include header.

| Elements | Description |
| --- | --- |
| Caption | It is the wrapper element to include title and sub-title. |
| Image | It supports to include header images with the specified dimensions. |

| Class | Description |
| --- | --- |
| e-card-header-caption | To group the title and subtitle within the header which acts as wrapper. |
| e-card-header-title | Main title text with in the header. |
| e-card-sub-title | A sub-title within the header. |
| e-card-header-image | To include heading image within the header. |
| e-card-corner | To add rounded corner for the image. |

Content :

Content in Card holds texts, images, links and all possible HTML elements. Its adaptable within the Card root element.

* Create a div element with the class e-card-content.
* Place content div element in the Card root element or within any Card inner elements.

[Class-component] .

7) How to install React Js on Windows, Linux Operating System? How to Install NPM and How to check version of NPM?

->

**React JS** -React is an open-source component-based front-end JavaScript library. It is used to create fast and interactive user interfaces for web and mobile applications. It is easy to create a dynamic application in React because it requires less coding and offer more functionality. It is used by big MNC and fresh new startups

### **Features of React:**

1. **Reusable Components**: A single React app consists of many components each component have their own logic and code  but we can easily reuse components any number of time hence reducing the developers time and increasing the efficiency of work
2. **Debugging**: React app can be easily debug using  “react developer tools”.It’s a browser extension that can be used for both chrome as well as Firefox.

### **Installation Reactjs on Windows:**

**Step 1**: Install Node.js installer for windows. Click on this [link](https://nodejs.org/en/). Here install the LTS version (the one present on the left). Once downloaded open NodeJS without disturbing other settings, click on the **Next**button until it’s completely installed.

**Step 2**: Open command prompt  to check whether it is completely installed or not type the command –>

node -v

**Step 3**: Now in the terminal run the below command:

npm install -g create-react-app

create-react-app --version

**Step 4:**Now Create a new folder where you want to make your react app using the below command:

mkdir newfolder

cd newfolder (your folder name)

**Step 5**: Now inside this folder run the command –>

create-react-app reactfirst YOUR\_APP\_NAME

**Step 6**: Now open the IDE of your choice for eg.  Visual studio code and open the folder where you have installed the react app **newfolder**(in the above example)  inside the folder you will see your app’s name **reactapp**(In our example). Use the terminal and move inside your app name folder.Use command  **cd reactapp**(your app name)

**Step 7:**To start your app run the below command :

npm start

npm version:

**npm –version**

**npm -v**

8) How to check version of React Js?

->

To check which React version is your project using you need to open the **package.json**. Take a look under the **dependencies** section. It should list all of the dependencies of your project and one of those should be React.

## Updating React version

To change the React version your project is using you need to run the following command  
**npm install –save react@<version>**, where **<version>** is a variable. You should substitute it for the desired version of React.  
Let’s take a look at how to downgrade it. I will instal the version 16 of React by running

$ npm install –save react@16

Now you can see that my **dependencies** section of the **package.json** has been successfully updated.

9) How to change in components of React Js?

->

### setState()

setState(updater[,callback])

setState() enqueues changes to the component state and tells React that this component and its children need to be re-rendered with the updated state. This is the primary method you use to update the user interface in response to event handlers and server responses.

The first argument is an updater function with the signature:

(state, props) => stateChange

state is a reference to the component state at the time the change is being applied. It should not be directly mutated. Instead, changes should be represented by building a new object based on the input from state and props. For instance, suppose we wanted to increment a value in state by props.step:

this.setState((state, props) => {

return {counter: state.counter + props.step};

});

Both state and props received by the updater function are guaranteed to be up-to-date. The output of the updater is shallowly merged with state.

The second parameter to setState() is an optional callback function that will be executed once setState is completed and the component is re-rendered. Generally we recommend using componentDidUpdate() for such logic instead.

You may optionally pass an object as the first argument to setState() instead of a function:

setState(stateChange[, callback])

This performs a shallow merge of stateChange into the new state, e.g., to adjust a shopping cart item quantity:

this.setState({quantity: 2})

10) How to Create a List View in React Js?

11) Create Increment decrement state change by button click?

->

import React from "react";

class StateEx extends React.Component

{

    //when you work with object in state than always use constructor.without that cant change value.

    //state={count:16}

    constructor()

    {

        super();

        this.state={count:1}

    }

    inc =()=>

    {

        // this.setState({count:2})

        //this.setState({count:this.state.count+1})  //ths.state.count =1 means 1+1

        // this.setState(()=>(

        //     {count:this.state.count+1}

        // ))

        this.setState((prev)=>(

            {count:prev.count+1}

        ))

    }

    dec =()=>

    {

        //this.setState({count:2})

        this.setState((prev)=>(

            {count:prev.count-1}

        ))

    }

    change =()=>

    {

        this.setState(this.state="Sahil")

    }

    render()

    {

        return(

            <>

            <button onClick={this.dec}>Decrement</button>

            <h1>{this.state.count}</h1>

            <button onClick={this.inc}>Increment</button>

            </>

        )

    }

}

export default StateEx;

Output:

