REACT JS

INTRODUCTION

React is java script library for create UI components.

React is use to create single page application.

React JS develop by Facebook.

React Use virtual DOM for generate DOM.

Developed by Jordan Walke.

WHY REACT IS VERY IMPORTANT

Maintained by Facebook so long term support available.

Highly demanded due to fast speed.

Large Community Support.

Mobile application development possible with React Native

First release on 29 may 2013.

Current version is 18

Big Customers

Whatsapp

Airbnb

Instagram

netflix

REQUIREMENTS

HTML

CSS

JS + ES 6 Concepts

INSTALLATION

We can install React JS with NPM

and also use React JS with CDN.

EXAMPLE 1

Create simple example of react with CDN.

```
<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
```

CDN REACT JS EXAMPLE

Open text editor.

Create simple .html page.

Add CDN in head section of page.

Add script tag with type = ‘text/babel’

Create simple class and extend React Component

Create render()

EXAMPLE CODE

```
<script type="text/babel">
        class Example extends React.Component{
                render(){
                        return <h1>Hello React JS</h1>
                }
        }
</script>
ReactDOM.render(<Example />, document.getElementByID(divid));
```

INSTALL NPM

Install Node

Install NPM

Install Cra

Verify versions of Node and NPM

Go to drive create folder for store projects.

Like reactProjects

Go to newly created folders,

Run commnd

Npx create-react-app app1

Open project in text editor software

Npm start to run project.

FIRST PROJECT

Create new js file and add in index.js

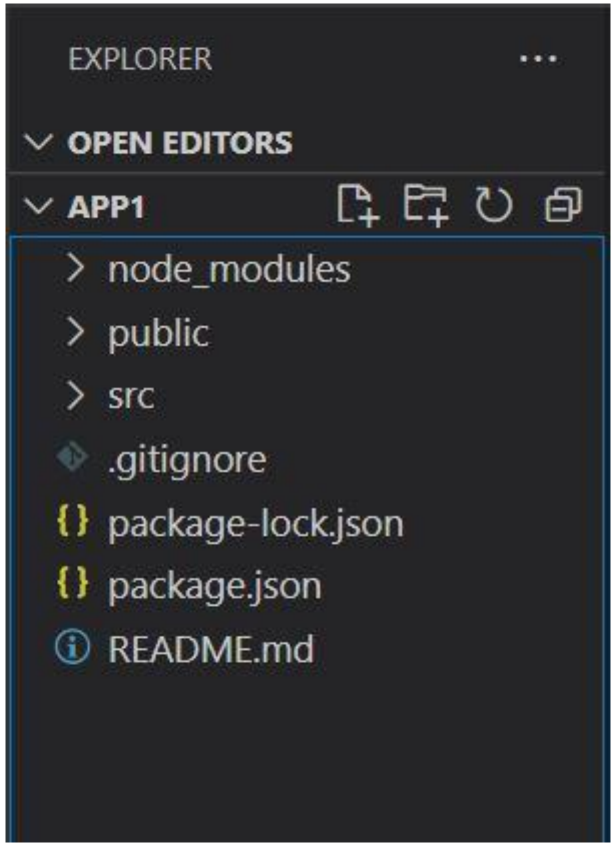
Do some code and execute it.

Alternative of NPM is YARN.

*YARN developed by Facebook.

FILE AND FOLDER STRUCTURE

Let we explore project structure of React JS.



```
node_modules
public
src
.gitignore
package-lock.json
package.json
README.md
```

PACKAGES.JSON

Centre point of our application

Contains all the information about our application like version, packages, files and etc.

PACKAGES-LOCK.JSON

History of packages file with resolver and dependency.

Use to store details about entire application structure.

SRC FOLDER

All the custom files contains in src folder with CSS and testing codes.

Index.css for style

Index.js entry point of react js

Logo application of logo

Report file

PUBLIC FOLDER

Icons and other static pages are available in public folder

NODE MODULES FOLDER

Contains all the packages of react js

BUILD FOLDER

Contains final project build

*created at end of the project.

PACKAGES.JSON IN DEPTH

Most important file in your React.JS Project.

Packages.json file contains

Packages.JSON file Contains

1. Versions.
2. Name,
3. Library,
4. Dependency,
5. Commands and etc.

ADD NEW PACKAGE IN APPLICATION

You can install packages like following

Npm install react-package-name

Example

Npm install react-validation

Or

You can add dependency in packages.json file

Example

In dependency section of packages.json file

```
{
        paste the module name with version
}
```

And run npm install command.

REMOVE PACKAGES

Npm remove package-name

Or

Remove from dependency section of packages.json file and run npm install command.

CUSTOM COMMANDS

In scripts section

And some command like :

“command name” : command

Like

“IANT” : ‘react-scripts start’

To run : npm run IANT

COMPONENTS IN REACT.JS

WHAT IS COMPONENTS

Component is a part or element of large project.

Like tyre is component of CAR

Battery is component is Mobile phone.

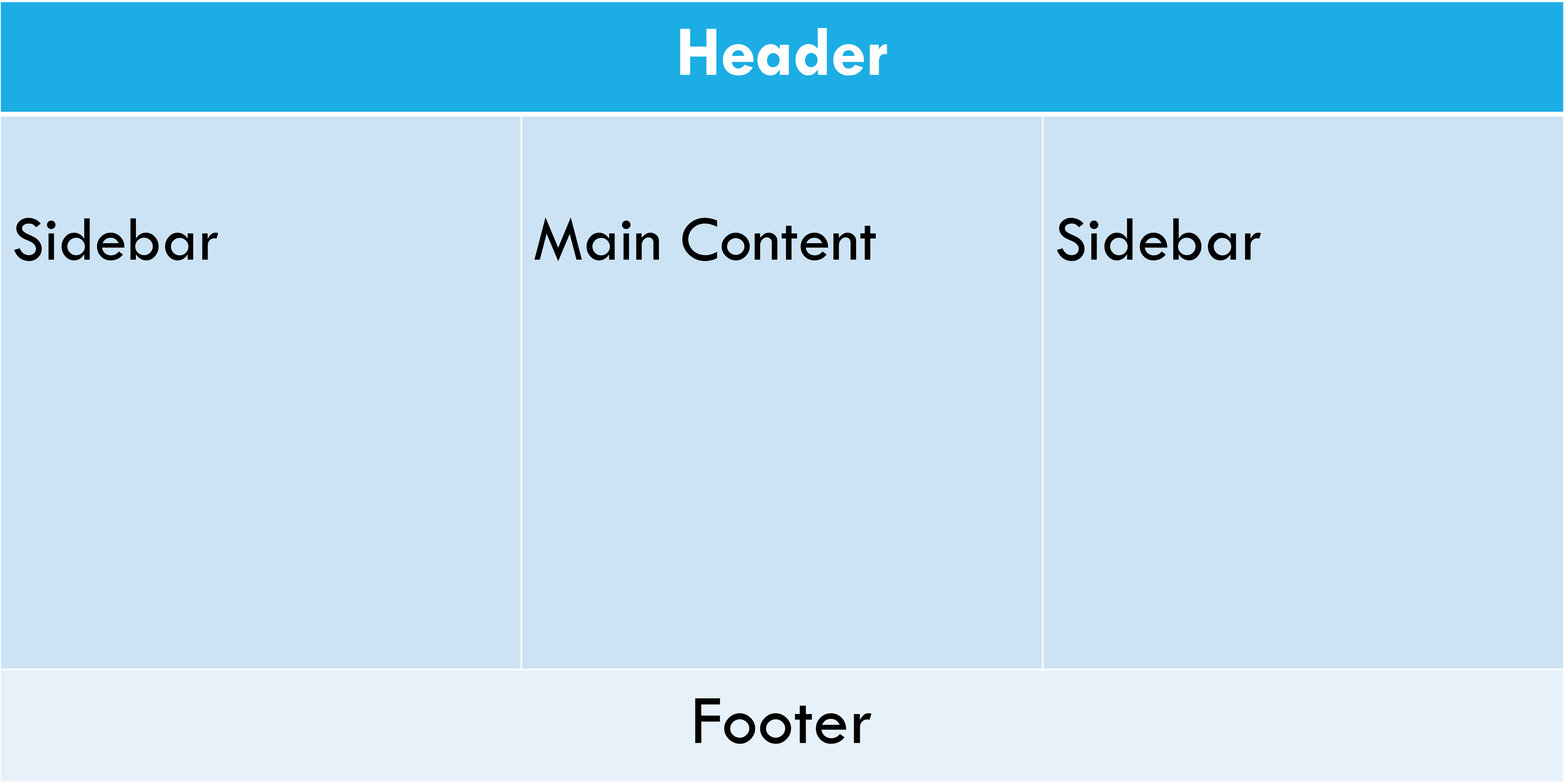
IN REACT WHAT IS COMPONENT

Block of code we can use many time as like function.

Every component have own life cycle.

Component is more powerful then functions because components have own html code and some advance concepts to pass parameters.

Header, menu and footer is best reusable component for all pages.


Header
Sidebar
Main Content
Sidebar
Footer

TYPE OF COMPONENT

Main two type of components

1. Class based component
2. Function based component

Sub category

1. High order component
2. Pure Component
3. Controlled Component
4. Uncontrolled Component
5. Nested Component

You can create multiple component in same file or you can create different files for different components.

* return only one tag from component.

You can wrap multiple component in single container.

You can remove “export defaults component-name” from last line of code if you add exports keyword before the function name.

* you need to import like this when you remove “export defaults component-name”

Import { function-name } from ‘./component-file’

You can create many components in App.js File they no need to import and export.

CLASS COMPONENT

Create some components in src.

Create some class for component

```
Class component name extends React.component{
}
```

You need to import component like following

```
Import React from ‘react’;

Class component name extends React.component{
}
```

```
Import React,{Component } from ‘react’;

Class component name extends component{
}
```

Every class component required render function to return some data

```
render(){
        return (some code )
}
```

*must export class from component.

*maybe class components deprecated in next few years by faceBook.

PRACTICE

Create class component in App.js file and use it.

NESTED COMPONENT

Create some component in component function.

You can use nested components in parent component only.

JSX

Java script XML is know as JSX.

Enable us to use JS code without use of <script> tag.

JSX give power to combine JS and HTML code.

JSX convert JS to XML

WITHOUT JSX

```
Import React from ‘react’

Function function_name()
{
        return React.createElement(“element Name”, “class”, “content”);
}
```

EVENTS AND FUNCTIONS

Create some function in App.js file

Create button to call newly created function.

You noticed the function is called without clicking button and when click on button is not called.

To prevent this issue remove parenthesis from function in event.

Use arrow function in event when need to do some alert message on event.

You can also use arrow function to call function with event.

Create some variable with data.

You can print variable data in js file with { variable name } format.

Change the value of variable in function and print it.

Data not updated after change value via function.

STATE IN FUNCTIONAL COMPONENT

We can use state in functional components as well as class components.

State is object in React to store some data.

We can use state instead of variable.

When we use variable, variable data is not updated with java script code.

React update component when changes detected in state or props.

TO CREATE STATE

Import usestate from React.

Declare state / destruct state

Const [data, setData]= usestate(“value”);

Print it with { data }

To change use setData = new value

TASK

Create function to use state with number value and increment with button click.

STATE IN CLASS COMPONENT

Create class component.

Create constructor in class component

Call super from constructor.

Like following

```
Class componentName extends React.Component{
        constructor(){
                super();
                this.state={
                        data:value
                }
        }
        render(
                return()
        )
}
```

TO USE STATE

{this.state.data} in html render code.

Try to update value of class component state like following

```
<button onClick=(()=>this.functionName())>Update State</button>
```

To update value create function and update state as follows.

```
this.setState(name:new value)
```

PROPS WITH FUNCTIONAL COMPONENT

Poprs. use to pass some data in component

Props is same as like parameters.

Props full form is properties.

Main use to pass dynamic data to component.

Create one functional component in your project.

Return div and some text for function and use in your application.

You can pass props values in component like following.

```
<ComponentName propsName={data} />
```

To access props in component JS use

```
Function ComponentName(data){
}
```

Use following for print in component

```
Function Component_name(propName){
          return <h1>{propName.data}</h1>
}
```

To send multiple values

```
<Component_Name propsName={data1} propsName={data2} />
```

To Apply style on Return element use Following

```
Return <div style{{ property:value}}>
        Data
</div>
```

To send multiple data in one property of props.

```
<Component_name data={{key1:value1, key2:value2, key3:value3}}/>
```

CHANGE PROPS WITH EVENT

Import useState in application

Create use State for your Data

const [data, setData] = useState(“Default Value”);

<button onClick=“{()=>{setData(“New Data”)}}”></button>

PROP WITH CLASS COMPONENT

Create basic application

Create Some component with class. Props are use to transfer some data between two or more components.

```
Class Component_name extends React.Component
{
        render(){
                return(
                        <div>Text</div>
                )
        }
}

export default Component Name
```

Use component and pass some data

<component Name data=“value”/>

TO USE PROPS IN CLASS COMPONENT

In render() of your application

{this.props.data}

CREATE EVENT TO CHANGE VALUES IN PROP

Note : prop values are updated in only component when we are sending the values, react not allowed to change values of props at receiving component.

Task : change prop values via class component.

WORK WITH INPUT

Get input box values in react.

Create component and return input type text from it.

Add on change event and call some function to get Data with receive some values in parameter.

Print data from parameter like following.

Parameter.target.value;

Create some state and save input values in state and go for further use as per your application requirement.

TASK

Create application to get value of input when user click on button.

Some state ?

If true :

If false

HIDE AND SHOW ELEMENTS

Hide some content or toggle content is basic need in front end application.

Create two buttons for hide and show content.

Create state with true value.

Change content of page with

{

  ternary operator

  condition ? True : False

}

TOGGLE BUTTON

To create toggle button create button and pass Boolean with not sign.

FORM HANDLING IN REACT

Create simple component with return some form

Add some input elements and submit button in form.

* not required to create from tag you just submit form data.

Add some function on onSubmit event of our form.

Create function in application with receiving event parameter.

Add event.preventDefault(); to function for prevent form submission.

create States for store form data.

Use useState functions for update data of our states when user change the value of input field.

Must pass event while calling function on event

Like : event.target.value;

event.target.checked;

Print all data on form submit.

TASK

Create clear button and set some default values in form.

CONDITIONAL RENDERING

Apply if else I React.

Create some component and apply with state.

We have two ways to render a conditional rendering.

Use if else for conditional rendering our component.

Use

```
If(state variable){
        return some code
}else{
        return some code
}
```

Use ternary operator for conditional rendering our component.

```
{
        state variable ? True Component : False Component
}
```

TASK

Implement If elseif in conditional rendering with ternary operator.

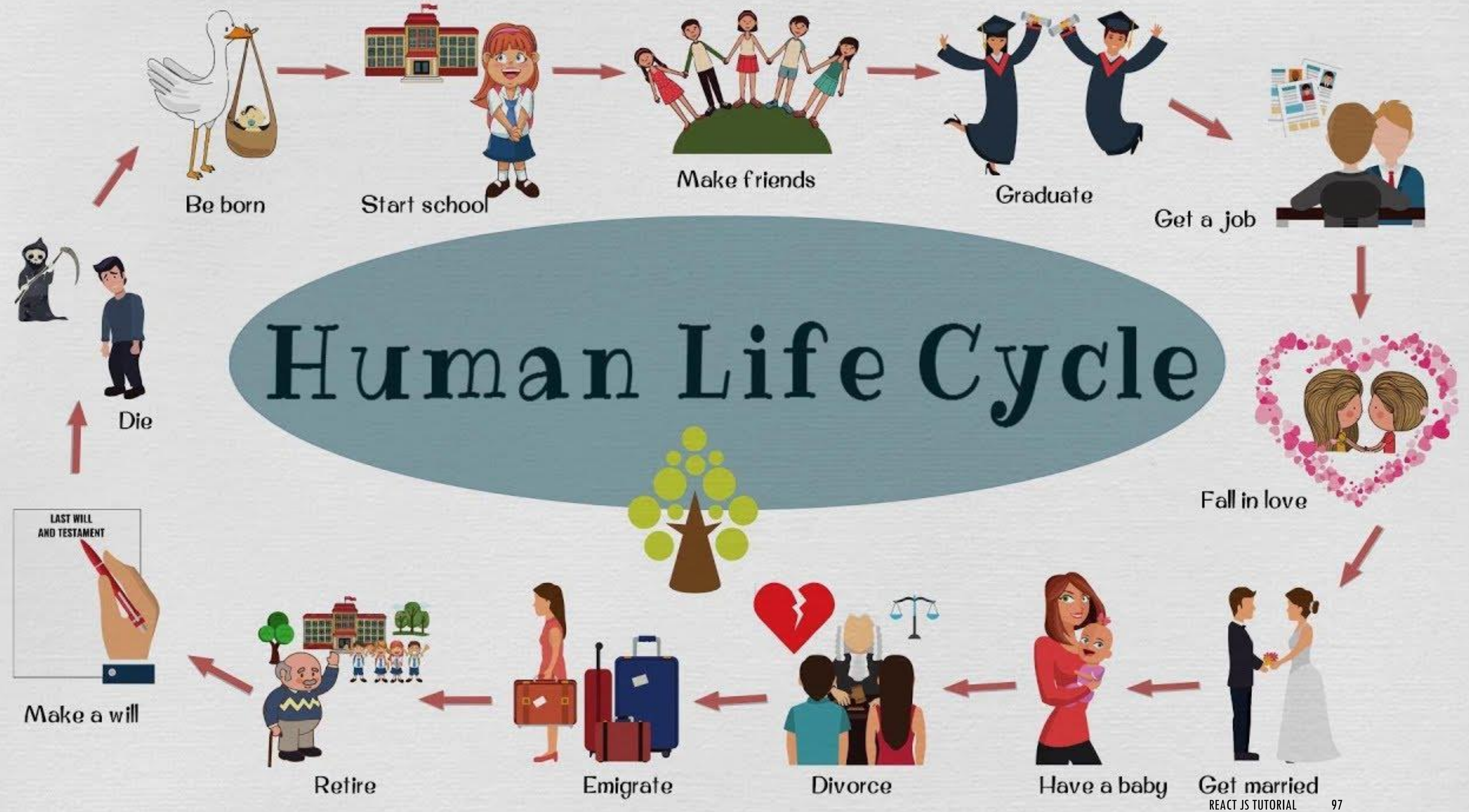
PASS FUNCTION AS PROPS

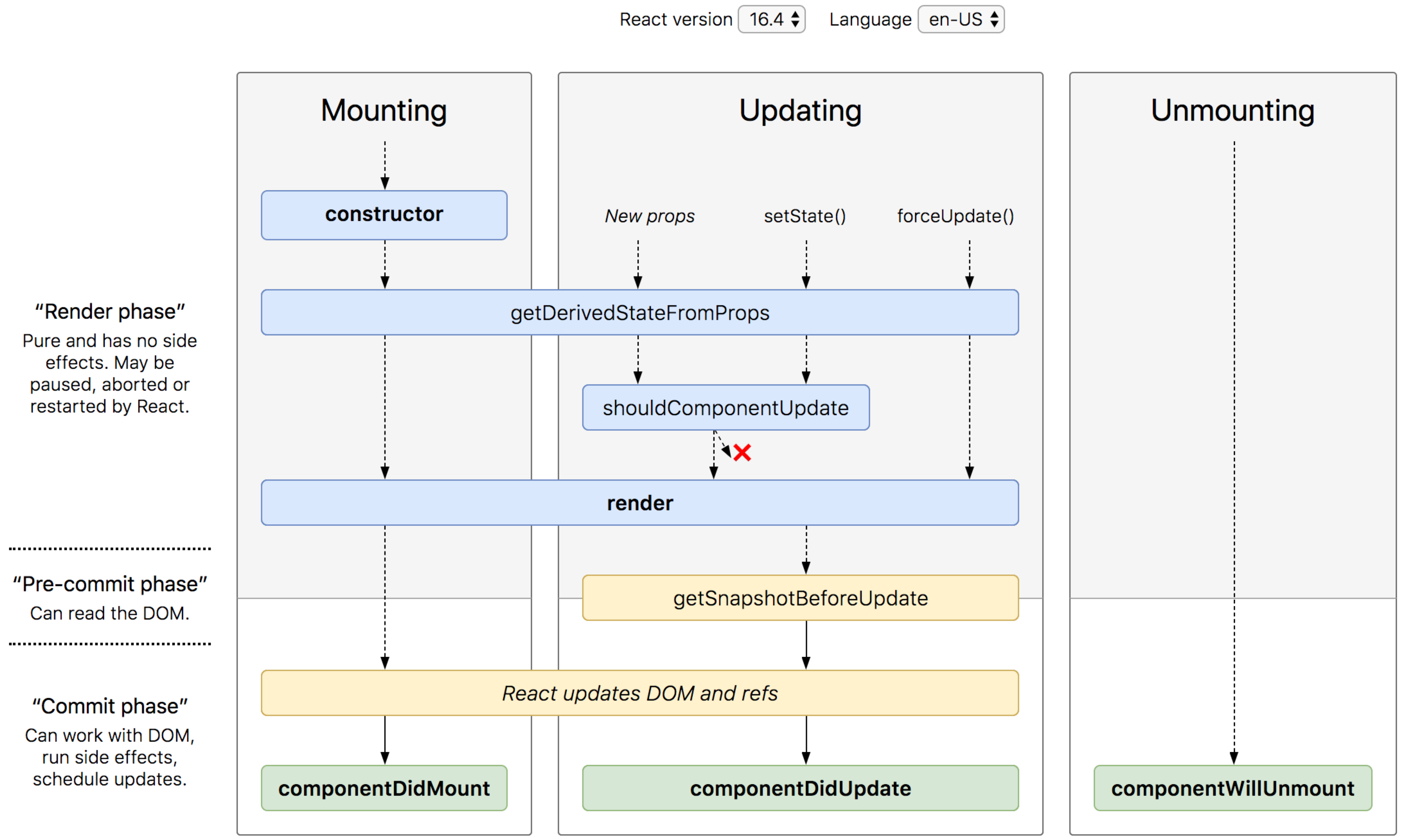
Create some component and import in parent component.

Create some function in parent component and pass the newly created function to our child component with use of props

Call the parent component function from child component with props.

LIFE CYCLE


Human Life Cycle
Be born
Start school
Make friends
Graduate
Get a job
Fall in love
Get married
Have a baby
Divorce
Emigrate
Retire
LAST WILL AND TESTAMENT
Make a will
Die


React version 16.4
Language en-US
Mounting
Updating
Unmounting
constructor
New props
setState()
forceUpdate()
getDerivedStateFromProps
shouldComponentUpdate
render
getSnapshotBeforeUpdate
React updates DOM and refs
componentDidMount
componentDidUpdate
componentWillUnmount
"Render phase"
Pure and has no side effects. May be paused, aborted or restarted by React.
"Pre-commit phase"
Can read the DOM.
"Commit phase"
Can work with DOM, run side effects, schedule updates.

PHASE OF LIFE CYCLE METHODS

Mount

Update

Unmounting

Create some component

When component is load mount phase is called.

When update some state or props update phase is called.

When remove component unmount phase is called.

Life cycle method called when component is load, update or unmouted.

CONSTRUCTOR LIFE CYCLE IN REACT

Class component with life cycle methods.

Constructor is called first when class component is created.

Constructor is ready before rendering your HTML.

Create state in constructor.

like this.state { state:value, stateN:valueN}

Must use super() in first line of your constructor.

Don’t call API from constructor.

RENDER LIFE CYCLE METHOD

Render meaning represent some in HTML form.

Render method used in class component only.

Write all HTML call in render while creating component with class component.

Render method runs when your component is ready.

Render method rerender HTML code when any state or props are get updated by your code.

Create some class component and pass data from parent component via parent component update state value and pass data with props.

COMPONENT DID MOUNT METHOD

This function called after all HTML, CSS (DOM) code ready.

Props and const update not effect on this method.

Must used when we use API.

Create some class component.

Create constructor method and call super()

Create componentDidMount() print some message

COMPONENT DID UPDATE METHOD

Called when component state or props are updates.

Create component with some state and update on button click.

Check for Componentdidupdate method.

* never update state in Componentdidupdate without if condition.

You can also use previous state

SHOULD COMPONENT UPDATE METHOD

In this life cycle method you can set how component state get update on some condition.

Use to stop specified rendering.

By default this method block whole rendering process.

Must return true if you want to update component.

Create a class component with some state to counter and set initial value to zero.

Increase counter value on button click.

Create life cycle method and make console.

COMPONENT WILL UNMOUNT METHOD

Component will unmount method called when our component get unmounted from our application DOM.

When component hide and show component completely unmounted from DOM.

Use for prevent / stop continue process like cancel API Call, close timer and etc.

HOOKS

To use life cycle methods in functional component we use hooks in our React application.

Class component have inbuilt features but functional component have no any extra features so we use hooks to get external functionality of class in our functional components.

Best example is state

Import hooks in our functional components.

Import React, {useState} from ‘react’

You must need to restructure hooks to use in functional component

* hooks start with use keyword

* use keyword is reserved to Hooks in React

* useEffect is used to implement life cycle methods in functional components.

USE EFFECT HOOK

Use effect used when you want to use

Component did mount,

Component did update,

Component will unmount methods in your functional component.

Create functional component

Import React from react

In function

```
React.useEffect(
        ()=>{
        }
);
```

Or

Import React, {useEffect} from react

In function

useEffect(()=>{

});

Use multiple useEffect in single functional for different states and props.

USE EFFECT HOOK WITH STATE & PROPS

You can use useEffect on specific state and props.

Create functional component with 2 states.

Create useEffect function

Increate value of first state on button click

Increate value of second state on button click

```
useEffect(()=>{
        some code
}, [state name])

useEffect(()=>{
        some code
}, [props.name])
```

Create another component pass some props

Pass some props from parent and do same as state.

Task : run same useEffect on two different states.

STYLE TYPE IN REACT

Style use to design html more attractive.

Type of styles.

Inline CSS style

CSS with module

CSS file

BootStrap

NORMAL STYLE

Create some css file with required tags and apply on tags.

Before use this type of css you need to import it and apply with className attribute.

STYLE INLINE

Use style attribute with inline css

<tag name style={{}}>content</tag name>

MODULAR CSS

Create some css file with

Filename.module.css

Add some css code in it.

Import file where you want to apply style.

Ex : import style from ‘filename.module.css’

For use css apply like following : className = {filename.classname}

INSTALL BOOTSTRAP IN REACT

Bootstrap is one of the popular CSS / JS library for design responsive web page in very short time with attractive look of our web page.

HOW TO INSTALL

npm install react-bootstrap bootstrap

Or

npm install bootstrap

Import { button } from react-bootstrap

Add variant in Buttons.

ADD ALERTS

```
{[
    'primary’,  'secondary’, 'success’,  'danger’,  'warning’, 'info’,  'light’, 'dark',
   ].map((variant) =3> (
    <Alert key={variant} variant={variant}>
     This is a {variant} alert—check it out!
    </Alert>
   ))}
```

HANDLE ARRAY IN REACT

Map is most important function in react JS.

Use Map function instead of for loop in React JS.

Map function use to access array data one by one and you don't need to find length or count index and etc.

For loop is not supported in return statement.

Create some array with some data.

Access array with map function.

```
Like arrayName.map((dataName)=>{
        logic of your code.
});
```

WITH FOR

```
for(let I = 0; i<array.length; i++){
        logic of code
}
```

TASK

Create some array of object and print it in table of react-bootstrap.

LIST WITH BOOTSTRAP AND UNIQUE KEY

Create some array of object with 5 or more records.

* unique key error.

When react handle array of object they need to identify every records uniquely so react throws warning when you not specified any unique key with record (object of you array).

Without unique virtual DOM are not functionating properly.

```
Arraylist.map((data, keyValue)=>{
        key = keyValue
        Logic code
})
```

* you need to specify key for every record of array as like above

* html validation error

Above error indicating improper HTML code in your component.

Add some conditional rendering with turnery operator to show specific data which fulfil your requirements.

NESTED ARRAY

Create some nested array and print with nested map()

REUSE COMPONENT

Component is reusable in react.

Create component and reuse with loops.

Create new project.

Add some JSON data to print on page.

Create functional component and return detailed data table.

Add newly created component with loop and pass JSON data as props of component.

*If possible Create function in parent components only, otherwise function makes multiple copy as we called our child components and this issue build more memory load on RAM.

REACT FRAGMENT

React fragment is pattern for handle multiple element and component.

Component is part of react with multiple element (tags).

FIRST WAY FOR USE REACT FRAGMENT

```
<React.fragement>
        code.
</React.fragement>
```

SECOND WAY FOR REACT FRAGMENT

Import fragment from react and use directly like following.

```
<Fragment>
 code
</Fragment>
```

THIRD WAY FOR REACT FRAGMENT

You just need to put <> </> to use Fragment.

Main use of Fragment while dealing with child components.

Create some child component with return some HTML code.

Example of return columns from Component.

SEND DATA FROM CHILD TO PARENT COMPONENT

Also known as lifting state up.

This is a basic feature of React JS.

Create some component with returning some HTML code with button.

Import child component in App.js and pass some data in child component via props and print it.

Create some function in parent Component pass function in child component and call function from child component.

You can send some data from child component to parent component function when you calling parent function in child component.

Same way receive that data in parent component.

TASK

Create some nested component and send top level function to super child component.

PURE COMPONENT

Pure component is one of feature of React that used only with class component.

Pure component prevent re rendering of component.

Prevent re rendering by checking state values if found different values then render the page otherwise not execute render.

Create a class component

Create some state and set some default value for state to zero.

create button to update state and increase value of state.

Import pure component in your component.

Extend pure component instead of component on your class.

PROPS WITH PURE COMPONENT

Task for you.

USE MEMO HOOK IN REACT

useMemo Hooks is very useful when you want to use pure component in your functional component.

useMemo hooks is increase speed of our application by preventing unnecessary rendering of our components.

Create application.

Create some functional component with two states and set some default values.

Create buttons to update states

Import useMemo with some function and set condition for specified state.

So when we update specified state then our memo function will called otherwise memo prevent our function.

REF IN REACT JS

Never update HTML DOM directly.

In react for update DOM must use state or props.

Make less use of ref in react.

Ref used with class components.

Ref used to manipulate DOM with super powers to do all changes of DOM.

Ref used in some emergency time only when you want to do more complex task with your HTML DOM.

Create new react application.

Create new class component.

Create class constructor with super constructor.

Import createRef with import statement.

In render return some input field with text type.

Declare ref in constructor,

```
Constructor(){
        super();
        this.inputRef = createRef();
}
```

Apply new created ref on input field like as following

```
<input type=“text” ref=“inputRef”>
```

Print ref In on component did mount for testing purpose.

You can also put current value in ref

This value appears on your input field.

Task

Change some property of input with button click.

USE REF HOOK

Use ref hooks used to access ref in functional component.

Use ref hooks for modify input field values, DOM, style, validations and etc.

Create new application with input box.

Create button to modify data of input.

Create variable to initialize hooks.

```
Let inputRef = useRef(null);
```

Apply ref hook with ref attribute.

```
<input type="text" ref={inputRef} />
```

Create button to acceds useRef

Create some function called on click of button.

```
Function handleInput(){
        inputRef.current.value = “new value”;
}
```

Task

Set focus back to input when click on button.

Set some style for input field for input field.

FORWARD REF IN REACT

Send ref to child component from parent component with props.

When we use ref its used for only same component only.

Use forward ref when you want to modify HTML DOM of child component.

Create an application.

Create button in parent component,

Create new functional child component, add some input box in child component.

Import ref in parent component and initialize with variable and set data to null.

Create function to update input and call function from button.

Use ref attribute of child component and pass ref to child component.

Import forwardref in your child component.

Must wrap your child component with forwardref like following

Export default forwardRef(child component);

Add two parameters in child component (props, ref)

And now you can use ref in your child component.

*React says never manipulate HTML DOM Directly.

CONTROLLED COMPONENT

Controlled component contained some input fields or entire form.

When we control input fields with states its known as controlled component in react.

Create functional component and add input text filed add some default value in input field.

When you set some default values to field you cannot allow to modify text field values.

Create state for handle form input.

Add value attribute and value to state value.

```
<input type="text" onchange=((event)=>stateUpdateFunction(event.target.value))
```

You need to create states for different input fields.

*add default value attribute for set default value for input text.

UNCONTROLLED COMPONENT

Components handled directly by DOM or ref is known as uncontrolled components.

Create new application

With two input fields and one button within form.

Add onsubmit event on form submit and call some function to prevent default action of submit.

Create two ref to handle input fields.

Get form data on submit of our form.

*you can also handle input by document.getElementById() function also.

HIGH ORDER COMPONENT

Component take another component as props and return also component its known as high order component.

Create an application.

Create new function in your app.js file

Return div with button and print some counter with state and print it.

Add multiple copy of newly created function you can see all the functions work properly.

Now time to create another function for handling high order component.

ROUTING

In react JS Routing used navigate one page to another page.

You need to convert components to pages and add some links its known as routing.

* you need to install react router.

```
npm install react-router-dom
```

Router, Route, Link the main important concepts in routing.

EXAMPLE

Create two components with same file.

Like Home, About Component and return some data from component.

Import {BrowserRouter as Router} from ‘react-router-dom’

Wrap your components between <Router >

COMPLETE EXAMPLE

```
import {BrowserRouter as Router, Link, Route, Routes} from
'react-router-dom'
function App() {
return (
    <div>
        <Router>
          <Link to="/home">Home</Link> |
          <Link to="/about">About</Link>
<Routes>
            <Route path='/home' element={<Home />}></Route>
            <Route path='/about' element={<About />}></Route>
          </Routes>
        </Router>
     </div>
  );
}
```

```
function Home(){
  return <div>
    <h1>Home</h1>
    <p>Some Content</p>
  </div>
}
```

```
function About(){
  return <div>
    <h1>About</h1>
    <p>About Component</p>
  </div>
}
export default App;
```

ROUTING EXAMPLE – 2 BEST PRACTICE

Create new project.

Add Router Wrapper in index.js file

```
Import { BrowserRouter as Router } from react-router-dom

<Router>

</Router>
```

Add All the links in navigation file.

Import React from ‘react’

Import { Link } from ‘react-router-dom’

Create function to return navigation links

Export function

Import nav function.

Add navigation and run project.

* set default route

Just set router link to “/” and put this link to bottom of the navigation bar set exact={true}.

404 - PAGE

404 – page not found used when no route found in your application.

Create one basic react application.

Make some component with routing.

```
Import {BrowserRouter as router, Link, Route} from ‘react-router-dom’

<Router>
        <link to=“/”>Home</link>
        <link to=“/about”>About</link>
        <Route path=“/” exact={true}><Home/></Route>
        <Route path=“/about”><about/></Route>
</Router>
```

*create new 404 component.

Add new Route for 404.

Import switch from react.

Wrap all of the Route component within switch and add 404 error component.

<Route path=“*”><404 Component /></Route>

DYNAMIC ROUTING WITH PARAMETERS (OLD VERSION)

When you want to large number of routes its must required to implement dynamic routing.

Give id or name to every routes.

Create some list of users.

Create loop on users list to print usernames.

Inappropriate way

Add links <a> on every username while running loop. // not perfect way

Appropriate way.

Import { BrowserRouter as Router, link, Route } from react-router-dom

Convert Simple Links to React-JS Router

```
<Router>
 users.map(()=>
            <div><h1><Link to={"/users/"+users.id}> Username</Link></h1></div>
            )
</Router>
<Route path="/user/:id"><UserComponent/></Route>
```

Create component to receive user Data.

Import { WithRouter } from ‘react-router-dom’

Return some elements from users function.

export userComponent withRouter(userComponent)

*wrap user Component with withRouter to access parameter values.

*get parameters data.

props.match.params.parametername

* to send multiple data via parameters.

<Link to="/users/:id"+users.id+"/"+users.fname"> Link Text </Link>

<Route path="/users/:id/:fname">Users Component</Route>

DYNAMIC ROUTING NEW VERSION OF REACT-ROUTING V 6.0

Create all links as created in previous example.

Use useparam reference to get parameter values in child component.

```
 <BrowserRouter>
     {
       students.map((row)=>
           <Link to={"/users/"+row.roll+"/"+row.fname}>
           {row.fname}</Link>)
     }
     <Routes>
       <Route path='/users/:roll/:fname' element={
        <Users />}></Route>
     </Routes>
</BrowserRouter>
```

```
import {useParams } from "react-router-dom";

function Users(props){
     const {roll} = useParams();
     const {fname} = useParams();
     return <>
          <h1>Welcome {fname} Your Roll no. is {roll}</h1>
     </>
}
export default Users;
```

API

API stands for application programming interface.

React, Angular and other front end frameworks are not able to connect database directly.

We need some API from backend side languages like Node – JS, PHP, ASP, Python and etc.

Mostly API Exchange data in JSON format.

GET – method used to get data from API.

POST – method used for store some data via API.

PUT – method used for update data via API.

DELETE – method used to delete data via API.

GET METHOD

Get method used to get some data in react frontend from backend server technology like PHP, Node.

Use fetch method of react to get data from API.

Fetch function return promise on function Call.

IN ANY COMPONENT (NOT PROPER WAY)

```
function componentName(){
        fetch(“URL of API”).then((result)=>{
                result.json().then((response)=>{
                        console.log(“Received Response is ”, response);
                })
        })
}
```

USE EFFECT (PROPER WAY)

```
useEffect(()=>{
        fetch(“URL of API”).then((result)=>{
                result.json().then((response)=>{
                        console.log(“Received Response is ”, response);
                })
        })
}, [])
```

Store received data in state like following.

```
const [data, setData] = useState([]);

// in fetch method

setData(response)
```

Print all the data with map function on data.

JSON SERVER

JSON server provides free and dummy API for testing purpose.

```
npm install -g json-server
```

create folder

go to folder

```
json-server --watch filename.json
```

POSTMAN

Postman is software to test API.

Available for both

You can download or add chrome extention.

POST METHOD WITH API

Post method of API is used to store data via API.

Create new application.

Create form with react.

Create states to handle input data.

Bind every fields with appropriate state and events.

Create function to store user data.

Create object of states.

FUNCTION CODE

```
Fetch("URL of API", {
            method : "POST",
            headers : {
                        'Accept':'application/json',
                        'Content-Type' : 'application/json',
            },
            body: JSON.stringyfy(data)
}).then((result)=>{
            // result to json and print it.
})
```

DELETE METHOD OF API

To delete specific data with API we need to use delete method of API.

PRE FILLED FORM

Pre filled form is very important when you want to update some data of existing list.

Create new form or use existing form.

Get some data form existing API.

And print in created form.

UPDATE DATA WITH PUT

Create new application.

Get all data from API and print in table.

Create button to get data in form.

Create button to update API data with PUT method.

PREVIOUS STATE IN FUNCTIONAL COMPONENT

Previous state used for get previous values of our state (value of state before update).

To find or calculate current state value and last state value previous state concept is very import.

PROGRAM

Create new app in react.

In app component create simple state with default value 1 and create button to update counter value.

Update random value in counter.

```
setCount((previous)=>{
        // access previous value with previous parameter
        // return new value to set in state
        // add some logic if you want to apply
})
```

ANOTHER EXAMPLE OF PREVIOUS STATE

Code same as like previous example.

Changes

```
for(let i=1; i<=5; i++){
        //setCount(count + 1);
        setCount((previous)=>previous + 1)
}
```

PREVIOUS PROPS

Use effect hooks is useful when you want to use previous props.

Create some child component and receive props from parent.

Create some state on parent and pass in child component props.

Create button in parent to send props in child component.

```
import React, {useEffect, useRef} from ‘React’;
function Users(props){
        const lastVal = userRef();
        useEffect(()=>{
                lastval.current = props.count
        })
        const previousProps = lastVal.current;
        return (<>
                <h1>{ props.count }</h1>
                <h1>{ props.count }</h1>
        </>)
}
```

CONTEXT API

When you want to communicate two component its required to context API is must required.

Context API is lightweight solution to transfer data to component to any component.

Create, Provider and consumer is most important part of context API.

Create folder named components.

Store all components in components folder.

Create file commonContext.js file to share data

COMMON FILE

Import react from react.

Export const Common = react.createContext();

Import common file in App.js

Create some common data in app.js

<common provider=“common data to share with all”>

<h1>Data</h1>

</common>

Create constructor and create state in constructor.

Create function to change colour of components.

Set some colour for component.

PROJECT BUILD

NPM RUN BUILD

When you complete your project development its time to move your project in production server. To build your project from react-js you need to run following command on your project folder.

```
npm run build
```

npm run build **creates a build directory with a production build of your app**. Set up your favorite HTTP server so that a visitor to your site is served index.html , and requests to static paths like /static/js/main.<hash>.js are served with the contents of the /static/js/main.<hash>.js file.

For testing purpose you need to install serve command like following

npm install –g serve

To run project

serve –s build

PROJECT 1

DESCRIPTION

Create weather details application with use of react and API

Design your page as you desired,

Get city name from user and print all weather details in output.

Requirement.

Get live data from API

Good GUI

GET WEATHER DATA FROM FOLLOWING LINK

https://openweathermap.org/api

Create account,

Verify account,

Get API key,

PROJECT 2

Create mysql database table with some product information.

Create API with php to fetch products by category.

Call API from your react project and display all product information on react.

* must use PHP & MYSQL as back end.

* must use routing in react.

* improve your front end design with bootstrap / react-bootstrap / .css code

PROJECT 3

Create some to do list and add new task for day.

You can edit or delete any task.

* must use PHP – MYSQL as backend.