**REACT Notes**

java script library for creating using interfaces it is not a framework but a library

-- component based approach

-- if we have to use a piece of code again and again we use react to create a fun type thing placee the code inside it and use that fun for that piece of code ( it will now be a component)

-- we can create component for samll piece of code as well as ;arge piece of coe and then call that component

<div>

<component/>

<div/>

--react uses declarative approach (client only have broad concept about the requirement but the developer build it for him)

--how much Javascript required for react?

it only depends upon what you are designing but learn basics for java script

--why react?

large github community and repository

--packagejson file

packagejson contain info about our app

favicon the small icon that appears in tab related to site

react dom to run html file

babel compiler for java script it turns our code such that browser understands it

**What is the DOM? The DOM (Document Object Model) represents the web page as a tree structure. Any piece of HTML that we write is added as a node, to this tree. With JavaScript, we can access any of these nodes (HTML elements) and update their styles, attributes, and so on.**

**Whenever u want to use javascript inside render method use curly brackets**

import React from 'react';

import ReactDom from 'react-dom'

// ReactDom.render('what to display','where to display','callback fun');

// 1 way

ReactDom.render(<h1>hello world</h1>,document.getElementById('root'));

/\* <h1>hello world</h1> \*/

// jsx expression

//react module is to be imported to use jsx expression

//react dom creates element for our h1

//2way -- ReactDom.render(React.createElement("h1",null,"hello world"),document.getElementById('root'));

//this is what happens orignally when we use jsx expression in reactdom.render this is what our browser sees

//3 way---pure java script code for the above work

// var h2=document.createElement('h1');

// h2.innerHTML='hello world';

// document.getElementById('root').appendChild(h2);

How to write multiple jsx elements

In this case in root tag in index.html

These tags are created h1 p h2

import React from 'react';

import ReactDom from 'react-dom'

ReactDom.render(

[<h1>hello world</h1> , <p>omago turu lub</p> ,<h2> second heading</h2>],

document.getElementById('root'));

//if the react version is above 16 we can array type structure can use [ ] brackets in start and end and use commas in between tags

// or jsx expressions

Another method

In this case in root div another div is create and in that div our tags are placed this may cause issue when applying css to cope with this we use fragments

import React from 'react';

import ReactDom from 'react-dom'

//render only take one element

//if we want to use multiple jsx elements in render method we have to wrap them up in single div tag

ReactDom.render(

<div><h1>hello world</h1>  <p>omago turu lub</p> <h2> second heading</h2></div>,

document.getElementById('root'));

React Fragment

It removes extra div node which was created so we use this to improve a bit efficiency

import React from 'react';

import ReactDom from 'react-dom'

ReactDom.render(

<React.Fragment>

<h1>hello world</h1>

 <p>omago turu lub</p>

 <h2> second heading</h2>

 </React.Fragment>,

document.getElementById('root'));

JAVASCRIPT EXPRESSIONS IN JSX

import React from 'react';

import ReactDom from 'react-dom'

const name="waleed";

//we have to use java script in html element

// which is done by using curly brackets

ReactDom.render(

<React.Fragment>

<h1>my name is {name} </h1>

<p>My lucky no  is {34.5\*2}</p>

<p>My lucky no  is {Math.random()}</p>

 </React.Fragment>,

document.getElementById('root'));

//we only use expressions in curly brackers

// can not use statements (if else for loops etc)

Template Literals in JSX

import React from 'react';

import ReactDom from 'react-dom'

const fname="waleed";

const lname="ishtiaq";

ReactDom.render(

<React.Fragment>

<h1>my name is {fname} {lname}</h1>

 </React.Fragment>,

document.getElementById('root'));

// output waleed ishtiaq

ReactDom.render(

    <React.Fragment>

    <h1>my name is {fname} +{lname}</h1>

     </React.Fragment>,

    document.getElementById('root'));

    //waleedishtiaq

ReactDom.render(

    <React.Fragment>

    <h1>my name is {fname} +" " + {lname}</h1>

     </React.Fragment>,

    document.getElementById('root'));

    //waleed ishtiaq

    //template literals

    ReactDom.render(

        <React.Fragment>

        <h1>my name is {`${fname} ${lname}`}</h1>

         </React.Fragment>,

        document.getElementById('root'));

        //or can use template litrals as

        ReactDom.render(

            <React.Fragment>

            <h1>{`my name is ${fname} ${lname}`}</h1>

             </React.Fragment>,

            document.getElementById('root'));

            // to get local date

            const currdate=new Date().toLocaleDateString();

           // to get your machine

            const currtime=new Date().toLocaleTimeString();

            ReactDom.render(

                <React.Fragment>

                <h1>{`date is  ${currdate}`}</h1>

                <h1>{`time is  ${currtime}`}</h1>

                 </React.Fragment>,

                document.getElementById('root'));

alt="random images"

Attributes in JSX

import React from 'react';

import ReactDom from 'react-dom'

const fname="waleed";

const lname="ishtiaq";

ReactDom.render(

<>

<h1 contentEditable="true">my name is {fname}</h1>

<img src="https://picsum.photos/200" alt="random images"/>

</> ,

document.getElementById('root')

);

//now we can edit on webpage E capital always

//<img/> is a self closing tag which we use here

//if our url of pic is changed by mistake or whatever the reason

//it will not appear on web page in this case the text written in

//will be displayed on screen

const img="https://picsum.photos/200" ;

const img1="https://picsum.photos/230" ;

const img2="https://picsum.photos/220" ;

const link="https://strikeout.ws/ufc-events-ultimate-fighting-championship-stream-1";

ReactDom.render(

    <>

    <h1 contentEditable="true">my name is {fname}</h1>

    <img src={img} alt="random images"/>

    <img src={img1} alt="random images"/>

    <a href={link} target\_="\_blank">

    <img src={img2} alt="random images"/>

    </a>

    </> ,

    document.getElementById('root')

    );

//now if we click on the third pic we will taken to

// the link mentioned in a tag

Css styling and importing

Classname keyword is used and we also need to import the index.css file

import React from 'react';

import ReactDom from 'react-dom'

import './index.css'

const fname="waleed";

const lname="ishtiaq";

const img="https://picsum.photos/200" ;

const img1="https://picsum.photos/230" ;

const img2="https://picsum.photos/220" ;

ReactDom.render(

    <>

    <h1 className='heading'>my name is {fname}</h1>

    <div className='imgdiv'>

    <img src={img} alt="random images"/>

    <img src={img1} alt="random images"/>

    <img src={img2} alt="random images"/>

    </div>

    </> ,

    document.getElementById('root')

    );

\*{

    margin:0;

    padding:0;

    box-sizing: border-box;

}

.heading{

color:brown;

text-align: center;

text-transform: capitalize;

font-weight: bold;

}

.imgdiv{

    display: flex;  /\* what it does is that in places all three images side by side in centre if we dont use this images will come one below another \*/

    justify-content: center;

}

.imgdiv img{

    width: 250px;

    height:300px;

    margin:70px,0px;

}

If you want to add or use google font add that link in index,,html file in header tag

And use that font in desired tag in css file by using name of that font in font-family

Internal css

import React from 'react';

import ReactDom from 'react-dom'

// import './index.css'

const fname="waleed";

const lname="ishtiaq";

const img="https://picsum.photos/200" ;

const img1="https://picsum.photos/230" ;

const img2="https://picsum.photos/220" ;

const heading={  //it is a object

    //key:'value'

    color:'#fa9191'  ,         //value always in quote

    textTransform:'capitalize',  //should use camel case (niceWord) instead of kebab case (nice-word)

    textAlign:'center'

}

ReactDom.render(

    <>

    {/\* first way of internal styling \*/}

    <h1  style={heading}>my name is {fname}</h1>

      {/\* second way of internal styling \*/}

    <h1  style={{color:'#fa9191'  , textTransform:'capitalize'}}>my name is {fname}</h1>

    <div className='imgdiv'>

    <img src={img} alt="random images"/>

    <img src={img1} alt="random images"/>

    <img src={img2} alt="random images"/>

    </div>

    </> ,

    document.getElementById('root')

    );

MINI PROJECT 1

PRINTING HELLO SIR GOOD MORNING/AFTERNOON.EVENING ACCCORDING TO TIME

import React from 'react';

import ReactDom from 'react-dom'

 import './index.css'

const fname="Hello sir ";

const currtime=new Date().toLocaleTimeString();

const page={

    backgroundColor:'#fa9191' ,

    height:'100%',

    width:'100%'

}

const heading={

    background:'#fa9191'  ,         //value always in quote

    textTransform:'capitalize',

}

let curDate=new Date();

curDate=curDate.setHours(17);

let greeting='';

const greet={

};

if(curDate>=1 && curDate<12)

{

    greeting='GOOD MORNING';

    greet.color='green';

}

else if(curDate>=12 && curDate<19)

{

    greeting='GOOD Afternoon';

    greet.color='orange';

}

else

{

    greeting='GOOD Night';

    greet.color='black';

}

ReactDom.render(

    <><div>

    <h1  style={heading}>{fname}<span style={greet}>{greeting}  </span> </h1>

</div>

    </> ,

    document.getElementById('root')

    );

body{

    background-color: aqua;

    margin:0%;

    padding: 0%;

}

div{

    width:100%;

    display:flex;

    justify-content: center;

    margin-top: 60px;

}

**Components in react**

We try to keep our index.js file as simple as possible so what we do is we create as much no of components as we need and then import them all in app.jsx file and then finally at last import that app.jsx file index.js file

What happens if you don't capitalize the first letter of a component name? There will be an ambiguity, whether it is an HTML tag or a React component.

**Index.js file**

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

 import Heading from './Heading';

import Para from './Para';

 import List from './List';

ReactDom.render(

    <>

        <Heading></Heading>

        <Para></Para>

        <List></List>

    </>,

    document.getElementById('root')

)

**Heading.jsx file**

import React from 'react';

function Heading()

{

  return(

  <h1>my name is waleed</h1>)

}

export default Heading;

**Para.jsx file**

import React from 'react';

function Para()

{

  return(

    <p>this is my world</p>)

}

export default Para;

**List.jsx file**

import React from 'react';

function List()

{

    return(

        <ol>

            <li> hassan</li>

            <li> hassan</li>

            <li> hassan</li>

        </ol>

    )

}

export default List;

**IF YOU WANT TO RETRUN MULTIPLE ITEMS FROM A COMMPONENT WARP THEM UP IN FRAGMENT AND THEN RETURN**

**RETURN{**

**<>**

**<div>**

**<h1>**

**</h1>**

**</div>**

**</>**

**};**

**IMPORT EXPORT MODULES**

If we have not created a fun in component and instead created

const name=”waleed”

Export default name

In this case we don’t return only use export and in the main file we access this by using curly bracket instead of tags around it

{name}

What we name used during import should be again used when we are using the component and this value is not necessarily same of what we used during exporting this happens when we exported using default

We can use default but only for one

Export default name

Import my from “./app”

{my}

or

Export default name

Import name from “./app”

{name}

**:** when exporting more than one

Export default one;

Export {two}

Import one, {two} from”./app.js”

If there is also a fun which we have to return then

Fun myname()

{

Let name=’waleed’

Return name;

}

Export default one;

Export{two,myname,,,,,,,,,}

We can export as much as we want and name them in bracket separated by commas and we also import by using their name in bracket

And we use the fun in our main file as

{ myname() }

We want to import all of them we write

Import \* as hello from “./app.js”

In this way a object is created of that component file and if we want to access what is inside the component we access using dot { hello.name} {hello.one} {hello.two} and default is accessed by

{hello.default}

**PROPS IN REACT**

Index.jsx

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

import Card from './Cards' ;

// if we simply call card we will get the same result but we need something which will help us in changing the name image title in card component

//  so we use props

// ReactDom.render(

//     <>

//    <Card></Card>

//    <Card></Card>

//    </>,

//     document.getElementById('root')

// )

// in this way we can assign values of our own by using the same fun again and again and sending a object props and then assigning values in the

// fun using props dot attribute name

ReactDom.render(

    <>

   <Card imgsrc=" img of season"

   title="A Netflix Original Series"

   sname="DARK"

   link=" link of season"></Card>

    <Card imgsrc=" img of season"

   title="A Netflix Original Series"

   sname="Breaking Bad"

   link=" link of season"></Card>

   </>,

    document.getElementById('root')

)

Card.jsx

import React from 'react';

import ReactDom from 'react-dom';

function Card(props)

{

    return ( <>

        <div className="cards">

            <div className="card">

                <img src={props.imgsrc} alt="myPic" className="card\_img"></img>

                <div className="card\_info">

                    <span className="card\_category">{props.title}</span>

                    <h3 className='card\_title'>{props.sname}</h3>

                    <a href= {props.link} target="\_blank">

                        <button>watch now</button>

                    </a>

                </div>

            </div>

        </div>

    </>)

}

export default Card

**Arrays in React**

We can create an array of objects in form of component and then simply export the array and then include the component in index.js file

Sdata.jsx file

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

import Card from './Cards' ;

// we are creating a array of objects each object contain info about the season

const Sdata=[

    {

    imgsrc:" img of season",

   title:"A Netflix Original Series",

   sname:"DARK" ,

   link:" link of season"

    },

    {

    imgsrc:" img of season",

   title:"A Netflix Original Series",

   sname:"Breaking Bad" ,

   link:" link of season"

    }

]

export default Sdata

index.jsx file

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

import Card from './Cards' ;

import Sdata from './Sdata'

ReactDom.render(

    <>

    <h1> LIST OF TOP 5 NETFLIX SERIES IN 2023</h1>

    <Card imgsrc={Sdata[0]. imgsrc}

   title={Sdata[0]. title}

   sname={Sdata[0]. sname}

   link={Sdata[0]. link}></Card>

    <Card imgsrc={Sdata[1]. imgsrc}

   title={Sdata[1]. title}

   sname={Sdata[1]. sname}

   link={Sdata[1]. link}></Card>

   </>,

    document.getElementById('root')

)

**IMPORTANT**

**If we want to give each season different color we will think of giving classname to each <card></card>**

**Thinking that we can apply css using the class name but keep in mind if we write className in component that component will consider className as a property like sname title etcnd we can not apply css using it so always keep this in mind.**

**So don’t use predefined keywords in react**

**MAP METHOD IN REACT**

**.** functional programming

Usually Three parameters value, index\_no, array we are working with

Calls the fun for each existing element in array

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

import Card from './Cards' ;

import Sdata from './Sdata'

// all our data is displayed using map fun if there were 3 4 5 season all will be displayed by using one card instead of 3 4 5 cards this is purpose

// map func

function ncard(val)

{

  return(

    <Card imgsrc={val. imgsrc}

    title={val. title}

    sname={val. sname}

    link={val. link}></Card>

  );

}

ReactDom.render(

    <>

    <h1> LIST OF TOP 5 NETFLIX SERIES IN 2023</h1>

    {Sdata.map(ncard)};

   </>,

    document.getElementById('root')

)

**SECOND METHOD**

import React from 'react';

import ReactDom from 'react-dom';

 import './index.css';

import Card from './Cards' ;

import Sdata from './Sdata'

// all our data is displayed using map fun if there were 3 4 5 season all will be displayed by using one card instead of 3 4 5 cards this is purpose

// map func

ReactDom.render(

    <>

    <h1> LIST OF TOP 5 NETFLIX SERIES IN 2023</h1>

    {Sdata.map(function ncard(val)

    {

    return(

    <Card imgsrc={val. imgsrc}

    title={val. title}

    sname={val. sname}

    link={val. link}></Card>

  );

    })};

   </>,

    document.getElementById('root')

)

**Fat Arrow Function**

function myname(waleed)

{

}

const myname=(waleed)=>{

}

Both are same

Benefit of fat arrrow

function myname(a,b)

{

return a+b;

}

const myname=(a,b)=> a+b;

ReactDom.render(

    <>

    <h1> LIST OF TOP 5 NETFLIX SERIES IN 2023</h1>

    {Sdata.map((val)=>

    {

    return(

    <Card imgsrc={val. imgsrc}

    title={val. title}

    sname={val. sname}

    link={val. link}></Card>

  );

    })};

   </>,

    document.getElementById('root')

)

Each index of array in Sdata file should have something unique so we add a attribute in each index called

Id and in CARD in index.js file we use the KEYWORD key always same spelling always and assign key value of id key={val.id}

 <Card imgsrc={val. imgsrc}

    title={val. title}

    sname={val. sname}

    link={val. link}

    key={val.id}>

    </Card>

Keys help react identify which items have changed added, removed, re-ordered to give a unique identity to every element inside the array a key is required