**REACT.js**

Introduction

React.js is a JavaScript library for building user interfaces.

Popular companies using react.js :

* Facebook
* Netflix
* Reddit
* Airbnd
* Code Academy

Single page applications

A single template is rendered at a single time. The template consists of different components. A component is a js class or function that returns a html or a jsx. Separate components are combined to a parent component.

Components

A component is an independent, reusable code block which divides the UI into smaller pieces.

*Function based components*

The first and recommended component type in React is functional components. A functional component is basically a JavaScript/ES6 function that returns a React element (JSX).

function Welcome(props) {

return <h1>Hello, {props.name}</h1>;

}

*(or)*

const Welcome = (props) => {

return <h1>Hello, {props.name}</h1>;

}

*Class based components*

The second type of component is the class component. Class components are ES6 classes that return JSX

class Welcome extends React.Component {

render() {

return <h1>Hello, {this.props.name}</h1>;

}

}

JSX

JSX stands for JavaScript XML. Instead of html we use JSX

Gives a much easier and understandable tags.

URL Routing

Multiple pages in are rendered is a single page by syncing the components using the Routers. React Router is a standard library for routing in React. It enables the navigation among views of various components in a React Application, allows changing the browser URL, and keeps the UI in sync with the URL.

Props

When there is a need to pass a a data from one component to another props can be passed through functions to component which can be utilised in the remaining child component and to the next component this process is called prop drilling.

the props are a type of object where the value of attributes of a tag is stored. The word “props” implies “properties”, and its working functionality is quite similar to HTML attributes. Basically, these props components are read-only components.

import React from 'react';

import ReactDOM from 'react-dom/client';

function Car(props) {

return <h2>I am a { props.brand }!</h2>;

}

function Garage() {

return (

<>

<h1>Who lives in my garage?</h1>

<Car brand="Ford" />

</>

);

}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<Garage />);

State

Is a js object used to represent the information of a particular component. We normally create a class based State now moderly We use HOOKS to create state using functions with getStates and setStates.

import React from 'react';

import ReactDOM from 'react-dom/client';

class Car extends React.Component {

constructor(props) {

super(props);

this.state = {

brand: "Ford",

model: "Mustang",

color: "red",

year: 1964

};

}

render() {

return (

<div>

<h1>My {this.state.brand}</h1>

<p>

It is a {this.state.color}

{this.state.model}

from {this.state.year}.

</p>

</div>

);

}

}

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

Component lifecycle

* Mounting phase – Components are added to the DOM
* Updating phase – Components are updated to the DOM
* Unmounting phase – Components are removed from the DOM

*Class components to perform the tasks*

* componentDidMount(){}
* componentDidUpdate(){}
* componentWillUnmount(){}

*Function components*

* useEffect(()=>{},[Dependencies])

HOOKS

Are functions used to manage States. Before HOOKS functional components cannot hold State.

Common HOOKS :

* useState() 🡪 set and update state
* useEffect() 🡪 performs effects in lifecycle

State management

Some states are to be managed globally for all the parent components to a root component by using Context API and Redux and without prop drilling.

Virtual DOM

* It is the virtual representation of the real DOM.
* The changes we make are made in the Virtual DOM and is rendered to the Real DOM. Only the updated DOM is rendered to the Real DOM.

Key prop

Keys help React identify which items have changed, are added, or are removed. Keys should be given to the elements inside the array to give the elements a stable identity. Mainly key prop should be unique. If key prop is not managed then we will get an error.

Event Listeners

Are the functions that performs an event when some action is taken on the component.

Some event listeners are:

* onClick
* onClickCapture
* onSubmit

Handling forms

There is no need of typing a long code lke in HTML rather we have states to maintain the input data.

import React from 'react';

import ReactDOM from 'react-dom/client';

function MyForm() {

return (

<form>

<label>Enter your name:

<input type="text" />

</label>

</form>

)

}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<MyForm />);

Conditional Rendering

Values rendered based on conditions like using

* If
* Else
* Else-If

and using different operators.

Common commands

npx create-react-app

npm start

npm run build