**React Important Things Taken From W3 Schools.**

**console.log("Hello from my side...");**

**console.log(" Some important things that must know before react.js");**

**Spread Operator:**

/\*

The JavaScript spread operator (...) allows us to quickly copy all or part of an existing

array or object into another array or object.

\*/

**Example: 01**

const numbersOne = [1, 2, 3];

const numbersTwo = [4, 5, 6];

const numbersCombined = [...numbersOne, ...numbersTwo];

**Example: 02**

const numbers = [1, 2, 3, 4, 5, 6];

const [one, two, ...rest] = numbers;

**Example: 03**

const myVehicle = {

brand: 'Ford',

model: 'Mustang',

color: 'red'

}

const updateMyVehicle = {

type: 'car',

year: 2021,

color: 'yellow'

}

const myUpdatedVehicle = {...myVehicle, ...updateMyVehicle};

**Notice the properties that did not match were combined, but the property that did match,**

**color, was overwritten by the last object that was passed.**

**The resulting color is now yellow.**

**Important Point Related To ES6 Modules:**

/\*

Import

You can import modules into a file in two ways, based on if they are named exports or default

exports.

1) Named exports must be destructured using curly braces.

2) Default exports do not.

Export

You can export a function or variable from any file.

There are two types of exports: Named and Default.

In-line individually:

(i) export const name = "Jesse"

(ii) export const age = 40

All at once at the bottom:

(i) const name = "Jesse"

(ii) const age = 40

export { name, age }

\*/

**Arrow Function: Sending the event object manually:**

function Football() {

const shoot = (a, b) => {

alert(b.type);

/\*

'b' represents the React event that triggered the function,

in this case the 'click' event

\*/

}

return (

<button onClick={(event) => shoot("Goal!", event)}>Take the shot!</button>

);

}

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

root.render(<Football />);

**Conditionals in React:**

function Goal(props) {

const isGoal = props.isGoal;

if (isGoal) {

return <MadeGoal/>;

}

return <MissedGoal/>;

}

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

root.render(<Goal isGoal={false} />);

/**React lists:**

Let's render all of the cars from our garage:

\*/

function Car(props) {

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

}

function Garage() {

const cars = ['Ford', 'BMW', 'Audi'];

return (

<>

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

<ul>

{cars.map((car) => <Car brand={car} />)}

</ul>

</>

);

}

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

root.render(<Garage />);

**map with unique id:**

function Car(props) {

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

}

function Garage() {

const cars = [

{id: 1, brand: 'Ford'},

{id: 2, brand: 'BMW'},

{id: 3, brand: 'Audi'}

];

return (

<>

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

<ul>

{cars.map((car) => <Car key={car.id} brand={car.brand} />)}

</ul>

</>

);

}

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

root.render(<Garage />);

**React Forms with useState Hook:**

import { useState } from "react";

import ReactDOM from 'react-dom/client'

function MyForm() {

const [name, setName] = useState("");

return (

<form>

<label>Enter your name:

<input

type="text"

value={name}

onChange={(e) => setName(e.target.value)}

/>

</label>

</form>

)

}

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

root.render(<MyForm />);

**Submitting Forms In React**

You can control the submit action by adding an event handler in the onSubmit attribute

for the <form>:

import { useState } from 'react';

import ReactDOM from 'react-dom/client';

function MyForm() {

const [name, setName] = useState("");

const handleSubmit = (event) => {

event.preventDefault();

alert(`The name you entered was: ${name}`)

}

return (

<form onSubmit={handleSubmit}>

<label>Enter your name:

<input

type="text"

value={name}

onChange={(e) => setName(e.target.value)}

/>

</label>

<input type="submit" />

</form>

)

}

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

root.render(<MyForm />);

**Textarea in React:**

import { useState } from "react";

import ReactDOM from "react-dom/client";

function MyForm() {

const [textarea, setTextarea] = useState(

"The content of a textarea goes in the value attribute"

);

const handleChange = (event) => {

setTextarea(event.target.value)

}

return (

<form>

<textarea value={textarea} onChange={handleChange} />

</form>

)

}

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

root.render(<MyForm />);

**Multiple Input Fields:**

1) You can control the values of more than one input field by adding a name attribute to

each element.

2) We will initialize our state with an empty object.

3) To access the fields in the event handler use the "**event.target.name**" and "**event.target.value**"

syntax.

4) To update the state, use square brackets [bracket notation] around the property name.

**Select in react:**

A drop down list, or a select box, in React is also a bit different from HTML.

in HTML, the selected value in the drop down list was defined with the selected attribute:

**HTML:**

<select>

<option value="Ford">Ford</option>

<option value="Volvo" selected>Volvo</option>

<option value="Fiat">Fiat</option>

</select>

**In React, the selected value is defined with a value attribute on the select tag:**

A simple select box, where the selected value "Volvo" is initialized in the constructor:

function MyForm() {

const [myCar, setMyCar] = useState("Volvo");

const handleChange = (event) => {

setMyCar(event.target.value)

}

return (

<form>

<select **value={MYCAR}** onChange={handleChange}>

<option value="Ford">Ford</option>

<option value="Volvo">Volvo</option>

<option value="Fiat">Fiat</option>

</select>

</form>

)

}