

1. In React, events are interactions that can be handled to trigger specific behavior in a component. These include user actions such as clicks, form submissions, mouse movements, keyboard input, and more. React provides a cross-browser wrapper around the browser’s native events, which ensures consistent behavior across different environments.

We attach events to JSX elements in a declarative manner, which allows us to define what should happen when a specific event occurs.

1. An event handler is a function that executes in response to a specific event. In React, we define event handlers as class methods or arrow functions (in functional components), and we bind them to specific elements using JSX.

Example:

function handleClick() {

alert('Button was clicked');

}

<button onClick={handleClick}>Click Me</button>

1. A SyntheticEvent in React is a wrapper around the native DOM event. It is part of React’s event delegation system, which uses a single event listener on the root to manage all events. This system improves performance and provides normalized behavior across all browsers.

React’s synthetic event system ensures that we interact with a consistent interface regardless of the differences in event implementations between browsers.

Example:

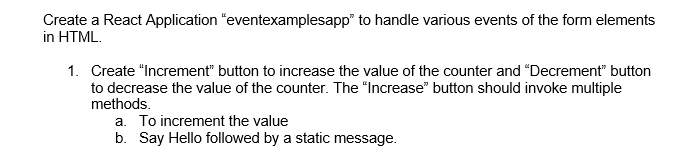
function handleInput(event) {

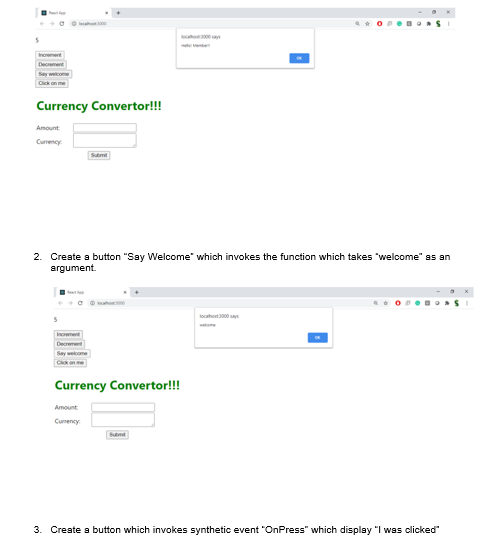
console.log(event.target.value); // event is a SyntheticEvent

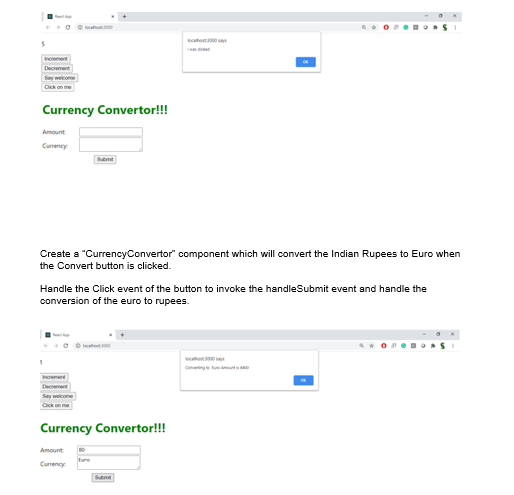
}

1. React follows a camelCase naming convention for event names. Unlike HTML where event names are lowercase (e.g., onclick, onchange), React requires us to write them as onClick, onChange, onSubmit, etc.

Furthermore, the value assigned to a React event must be a function reference, not a string.







**Solution**

**CurrencyConvertor.js**

import React, { useState } from 'react';

import './App.css';

function CurrencyConvertor() {

const [rupees, setRupees] = useState('');

const [euro, setEuro] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

const rate = 0.011; // Example: 1 INR = 0.011 Euro

const converted = (parseFloat(rupees) \* rate).toFixed(2);

setEuro(converted);

alert(`Converted Amount in Euro: €${converted}`); // dialog box

};

return (

<div>

<h2 className="cur">Currency Convertor!!!</h2>

<form onSubmit={handleSubmit}>

<label>Enter Amount in INR: </label>

<input

type="number"

value={rupees}

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

required

/>

<button type="submit">Convert</button>

</form>

{euro && <p>Converted Amount in Euro: €{euro}</p>}

</div>

);

}

export default CurrencyConvertor;

**App.js**

import logo from './logo.svg';

import React, { useState } from 'react';

import './App.css';

import CurrencyConvertor from './CurrencyConvertor';

function App() {

const [count, setCount] = useState(0);

// Increments counter and says hello

const handleIncrement = () => {

increment();

sayHello();

};

// Show alert instead of console log

const sayHello = () => {

alert("Hello Member!!");

};

// Increase counter value

const increment = () => {

setCount(prev => prev + 1);

};

// Decrease counter value

const handleDecrement = () => {

setCount(prev => prev - 1);

};

// Function with argument passed to event handler

const sayWelcome = (message) => {

alert(message);

};

// Synthetic event handler

const handleSyntheticClick = (event) => {

alert("I was clicked");

console.log(event); // SyntheticEvent object

};

return (

<div className="App">

<h2>Counter Value: {count}</h2>

<button onClick={handleIncrement}>Increment</button>

<button onClick={handleDecrement}>Decrement</button>

<br /><br />

<button onClick={() => sayWelcome("Welcome!")}>Say Welcome</button>

<br /><br />

<button onClick={handleSyntheticClick}>OnPress</button>

<br /><br />

<CurrencyConvertor />

</div>

);

}

export default App;

**App.css**

.App {

text-align: center;

}

.App-logo {

height: 40vmin;

pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

.App-logo {

animation: App-logo-spin infinite 20s linear;

}

}

.App-header {

background-color: #282c34;

min-height: 100vh;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

font-size: calc(10px + 2vmin);

color: white;

}

.App-link {

color: #61dafb;

}

@keyframes App-logo-spin {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

.App {

text-align: left;

padding: 20px;

}

button {

margin: 5px;

padding: 10px 15px;

font-size: 16px;

}

.cur {

color: green;

}

**Output**

