Event Handling

Introduction

Event handling in React is how you respond to user interactions like clicks, inputs, and key presses. It works similarly to vanilla JavaScript but uses **camelCase** for event names and functions instead of strings.

Key Points

- Events are written in camelCase (e.g., onClick, onChange).
- You pass a function reference, not a string (onClick={handleClick} not onClick="handleClick()").
- React automatically handles event binding, so you don't need addEventListener
- To call a function with parameters inside onClick, use an **arrow function**

Calling non-parameterized function

```jsx
function MyComponent() {
  const handleClick = () => {
    alert("Button Clicked!");
  };

  return <button onClick={handleClick}>Click Me</button>;
}
```

Calling Parameterized Functions

```jsx
function MyComponent() {
  const handleClick = (name) => {
    alert(`Hello, ${name}!`);
  };

  return <button onClick={() => handleClick("Alice")}>Click Me</button>;
}
```

This ensures the function is only called when the button is clicked.

Passing event Object

In React, you can pass the event object to a function by using an **arrow function** or passing it as an argument.

Automatically Passed by React

When an event handler is triggered, React automatically provides the event object.

```jsx
function MyComponent() {
  const handleClick = (event) => {
    console.log("Event:", event);
    alert("Button Clicked!");
  };

  return <button onClick={handleClick}>Click Me</button>;
}
```

Passing Extra Parameters with the Event

If you need to pass additional parameters along with the event, use an **arrow function**.

```jsx
function MyComponent() {
  const handleClick = (name, event) => {
    console.log("Event:", event);
    alert(`Hello, ${name}!`);
  };

  return <button onClick={(event) => handleClick("Alice", event)}>Click Me</button>;
}
```

Conclusion

Thus while implementing function calls on event remember following points

- Use an **arrow function** to pass extra parameters with the event.

- React automatically provides the event object, so you only need to handle it when necessary.
- Avoid direct function calls (onClick={handleClick()}) to prevent immediate execution.