📌 React Event Handling (이벤트 핸들링)

🚀 1. React에서 이벤트란?

📖 개념

- 이벤트는 사용자의 **클릭, 입력, 스크롤, 키보드 입력 등과 같은 상호작용**을 감지하여 특정 동작을 실행하는 기능입니다.
- React에서는 HTML 이벤트와 유사하지만 몇 가지 차이점이 있습니다.

✅ React 이벤트의 특징

1. 이벤트 이름이 **camelCase 표기법**을 사용 (`onclick` → `onClick`)
2. 이벤트 핸들러를 **문자열이 아닌 함수로 전달**해야 함 (`onClick={handleClick}`)
3. `return false` 대신 **`event.preventDefault()`를 사용**하여 기본 동작을 방지해야 함

```
function ClickButton() {
  function handleClick() {
    alert("버튼이 클릭되었습니다!");
  }

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

💡 2. React 이벤트 객체 (Event Object)

React에서 이벤트 핸들러 함수의 첫 번째 매개변수는 **SyntheticEvent(합성 이벤트)** 객체입니다.

```
function InputField() {
  function handleChange(event) {
    console.log("입력된 값:", event.target.value);
  }

  return <input type="text" onChange={handleChange} />;
}
```

✅ `event.target.value`를 사용하여 입력된 값을 가져올 수 있음

📌 이벤트 객체 활용 예시

```
function MouseTracker() {
  function handleMouseMove(event) {
    console.log(`마우스 위치: X=${event.clientX}, Y=${event.clientY}`);
  }
  return <div onMouseMove={handleMouseMove} style={{ height: "200px", backgroundColor:
"lightgray" }}>마우스를 움직여보세요</div>;
}
```

✅ `event.clientX`, `event.clientY`를 활용하여 마우스 위치 추적 가능

🛠 3. 클래스형 컴포넌트에서 이벤트 핸들링

클래스형 컴포넌트에서 이벤트를 사용할 때는 `this`를 바인딩해야 합니다.

```
class ToggleButton extends React.Component {
  constructor(props) {
    super(props);
    this.state = { isOn: false };
    this.handleClick = this.handleClick.bind(this);
  }

  handleClick() {
    this.setState(prevState => ({ isOn: !prevState.isOn }));
  }

  render() {
    return (
      <button onClick={this.handleClick}>
        {this.state.isOn ? "켜짐" : "꺼짐"}
      </button>
    );
  }
}
```

✅ `bind(this)`를 사용하여 이벤트 핸들러 내부에서 `this`를 정상적으로 사용할 수 있도록 해야 함

🔥 4. 함수형 컴포넌트에서 이벤트 핸들링 (Hooks 활용)

```
import { useState } from "react";

function ToggleButton() {
  const [isOn, setIsOn] = useState(false);

  return (
    <button onClick={() => setIsOn(!isOn)}>
      {isOn ? "켜짐" : "꺼짐"}
    </button>
  );
}
```

✅ **화살표 함수** 또는 **useState**를 활용하여 이벤트를 간결하게 처리할 수 있음

⚡ 5. 기본 동작 방지 (`preventDefault`)

기본적인 폼 제출 동작을 방지하는 방법입니다.

```
function FormComponent() {
  function handleSubmit(event) {
    event.preventDefault();
    alert("폼이 제출되었습니다.");
  }

  return (
    <form onSubmit={handleSubmit}>
      <button type="submit">제출</button>
    </form>
  );
}
```

✅ `event.preventDefault()`를 사용하면 **폼이 실제로 서버로 전송되지 않고 이벤트 핸들러 내부 로직만 실행**됨

🎯 6. 적용

✅ 이벤트 핸들러는 함수로 전달하여 실행

✅ 이벤트 객체(event)를 활용하여 입력값, 마우스 위치 등 처리

✅ 함수형 컴포넌트에서는 useState와 화살표 함수를 활용

✅ 클래스형 컴포넌트에서는 this 바인딩을 고려하여 이벤트 처리

✅ `preventDefault()`를 사용하여 기본 동작을 제어

🏁 7. 마무리

- React 이벤트 핸들링은 HTML과 유사하지만 camelCase 사용, 이벤트 객체 활용 등 차이점이 있음
- 함수형 컴포넌트에서 이벤트를 다룰 때 useState와 화살표 함수를 적극 활용
- 실무에서는 입력 폼, 버튼, 마우스 이벤트 등을 활용하여 동적인 UI를 구현 🚀