React Events - Detailed Explanation

# Explain React events

React events are the system React uses to handle user interactions such as clicks, key presses, form submissions, etc. React wraps the browser’s native events with its own system known as the Synthetic Event system to ensure consistency across different browsers.  
  
Unlike traditional HTML event handling where we use attributes like onclick, React uses camelCase syntax (e.g., onClick). These events are passed as functions and allow developers to define what should happen when the event is triggered.  
  
React events are declarative and bind directly in the JSX component code. For example:  
```jsx  
<button onClick={handleClick}>Click Me</button>  
```  
  
This will call the `handleClick` function when the button is clicked.

# Explain about event handlers

An event handler is a function that defines what happens when a certain event is triggered by the user. In React, event handlers are written as methods or functions and are passed to elements as props.  
  
Example of an event handler:  
```jsx  
function handleClick() {  
 alert('Button was clicked!');  
}  
  
<button onClick={handleClick}>Click Me</button>  
```  
  
The function `handleClick` is the event handler for the onClick event. Event handlers help make the UI interactive and dynamic by responding to user actions.

# Define Synthetic event

A Synthetic Event is a cross-browser wrapper around the browser’s native event system provided by React. It wraps around the native event and normalizes it so that it behaves consistently across all browsers.  
  
Synthetic events are part of React’s event delegation system which makes event handling more efficient by attaching events to a single root DOM node.  
  
Example:  
```jsx  
function handleInputChange(event) {  
 console.log(event.target.value); // event is a SyntheticEvent  
}  
```  
  
React reuses SyntheticEvent objects for performance, so their properties may be nullified asynchronously. Always use the event within the same synchronous call.

# Identify React event naming convention

React follows camelCase naming convention for event handlers rather than lowercase.  
  
Examples of naming conventions in React:  
- HTML: `<button onclick="handleClick()">`  
- React: `<button onClick={handleClick}>`  
  
Here are some more examples:  
- onClick  
- onChange  
- onMouseEnter  
- onSubmit  
  
Always remember to use the correct camelCase version of event names and pass functions directly (not strings).