REACT JS

Events

Sreekanth M. E.

Freelance Trainer & Consultant

http://www.SreekanthME.com

In React, regular DOM events are not exposed.

React-specific events of type SyntheticEvent have to be handled.

SyntheticEvent is React’s cross-browser wrapper around browser's native Event.

Event handlers don't get native event arguments of type MouseEvent, KeyboardEvent, etc.

They always get event arguments of type SyntheticEvent that wrap browser's native event.

SyntheticEvent that wraps the MouseEvent has access to mouse-specific properties.

SyntheticEvent that wraps the KeyboardEvent has access to keyboard-related properties.

DOM events and their properties may not map one-to-one with React events & their properties.

Some DOM events don't even exist in React.

React’s Events Documentation: https://facebook.github.io/react/docs/events.html

React components cannot have event handlers on them.

This is because components are after all just wrappers for DOM elements.

BENEFITS OF SYNTHETICEVENT :

- Browser Compatibility
- Improved Performance

BROWSER COMPATIBILITY

By wrapping all of the native events as an object of type SyntheticEvent, React avoids dealing with browser-specific event handling quirks.

IMPROVED PERFORMANCE

React manages events in a more efficient and performant manner than DOM.

React events are named using camelCase, rather than lowercase.

JSX uses a function as the event handler, rather than a string.

EXAMPLE OF DOM VS REACT EVENT HANDLING

```
<button onclick=“hitMe()">
  Hit Me!
</button>


<button onClick={this.hitMe}>
  Hit Me!
</button>
```

In DOM, an event’s default behavior can be prevented by just returning false.

In React, preventDefault() method has to be called explicitly.

A event handler is just a method on the component class.

In JS, class methods are not bound by default.

The event handler method has to be explicitly bound to this context. Else callback will be undefined when it is actually called in runtime.

There are different ways of binding an event handler to the this context.

COMPONENT INTERACTION

In React, Parent to child data flow is achieved using props.

To achieve child to parent data flow, handlers are passed into the child component from the parent component as props. The parent knows that such activity could occur in it's child which could trigger the handler passed by it.

When the handler method of parent is triggered, the handler can send new props to the child.

END OF CHAPTER

APPENDIX