Listening to Events and Working with Event Handlers

Responding to Events

- React lets you add event handlers to your JSX.
- Event handlers are your own functions that will be triggered in response to interactions like clicking, hovering, focusing form inputs, and so on.
- React is having default DOM behavior.
- To add an event handler, you will first define a function and then pass it as a prop to the appropriate JSX tag.
- If an element supports an **event**, then you can add a listener with React by adding such an **on** and then the event name **prop**.

For eg:

1. Declare a function called clickHandlers inside your Button component.

```
const clickHandlers = () =>{
    console.log("Clicked!!")
  } // function to run when button will be clicked.
```

2. Implement the logic inside that function(log the message on console).
3. Add onClick={clickHandlers} to the button JSX.

```
<button onClick={clickHandlers}>Change Title</button> <!-- event listeners to our elements>
```

- *"button"* is an Html element which supports event.
- *onClick* is an event listener.
- And then we just point at a function, either defined in line here or, better, defined upfront, and React will execute that function for you when that event occurs.
- *clickHandlers* is a function it will execute when event will occur.

> *Note:* {clickHandlers} is a function in curly braces without () paraentheses Because if we would add parentheses here, JavaScript would execute this when this line of code is being parsed.So it's then not executing clickHandler when the click occurs but when this JSX code is evaluated.

Important Functions passed to event handlers must be passed, not called. For example:

passing a function (correct) **calling a function (incorrect)**

```
<button onClick={handleClick}>        <button onClick={handleClick()}>
```

> The difference is subtle. In the first example, the handleClick function is passed as an onClick event handler. This tells React to remember it and only call your function when the user clicks the button.

> In the second example, the () at the end of handleClick() fires the function immediately during rendering, without any clicks. This is because JavaScript inside the JSX { and } executes right away.

```
import "./ExpenseItems.css";
import ExpenseDate from "./ExpenseDate";
import Card from "../UI/Card";

const ExpenseItems = (props) => {
  const clickHandlers = () =>{
    console.log("Clicked!!")
  } // function to run when button will be clicked.

  return (
    <Card className="expense-item">
    <ExpenseDate date={props.date}/>
      <div className="expense-item__description">
        <h2>{props.title}</h2>
      </div>
      <div className="expense-item__price">${props.amount}</div>
      <button onClick={clickHandlers}>Change Title</button> <!-- event listeners
to our elements>
    </Card>
  );
}

export default ExpenseItems;
```