Events in JS

The change in the state of an object is known as an Event

State change of an Object

Events are fired to notify code of "interesting changes" that may affect code execution.

- Mouse events (click, double click etc.)
- Keyboard events (keypress, keyup, keydown)
- Form events (submit etc.)
- Print event & many more

Event Handling in JS

```
node.event = ( ) => {
    //handle here
}
```

example

```
btn.onclick = ( ) => {
    console.log(“btn was clicked”);
}
```

Priority of Events
Events Handling in JS > Inline event handling in HTML


```
//Handle 1
let btn2 = document.getElementById("btn2");
btn2.ondblclick = () => {
    console.log("Double Click!");
    alert("Pressed double clicked button");
};

//Handle 2
btn2 = document.getElementById("btn2");
btn2.onmouseover = () => {
    console.log("Double Click!");
    alert("Pressed double clicked button");
};
```

If the same variable is updated then newly added event will overwrite the changes into the existing event.
--> In above example first Handle 1 had been executed.
In then handle 2 will overwrite the changes the handle 1 had made. So now onwards handle 2's statements are considered.

Event Object

It is a special object that has details about the event.

All event handlers have access to the Event Object's properties and methods.

```
node.event = (e) => {
    //handle here
}
```

e.target, e.type, e.clientX, e.clientY

Event Listeners

node.addEventListener(event, callback)

callback: It is a function that is used as an argument to the other function.
for ex. sum(), ele()
if it is used as sum(ele()); then the ele() function is called callback.

node.removeEventListener(event, callback)

*Note : the callback reference should be same to remove

for note, in short, we make a variable and store the callback func. into it. and then use it in both add and remove event listers.

```
ex. let handler = () => {
        console.log("clicked");
    };
    node.addEventListener("click",  handler);
    node.removeEventListener(event, handler);
```

Let‘s Practice

Qs. Create a toggle button that changes the screen to dark-mode when clicked & light-mode when clicked again.

toggle means,
if I have 2 states as 1 and 2
then going to state 2 from 1, going to state 1 from 2 and so on .....

HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Events and Handling</title>
    <link rel="stylesheet" href="/events.css">
</head>

<body>
    <!-- <button id="btn1">Works on Single Click</button>
    <button id="btn2">Works on Double Click</button> -->
    <button id="modeButton">Change Mode</button>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing
elit. Accusamus iste dolor ullam corporis architecto.
Accusamus quo consequatur ea ducimus repellat
consequuntur provident voluptate, tempore eveniet quod
enim distinctio doloremque. Deleniti.</p>

    <script src="/events.js"></script>
</body>

</html>
```

CSS

```
.dark {
    background-color: black;
    color: white;
}

.light {
    background-color: white;
    color: black;
}
```

JS

```
// let btn1 = document.getElementById("btn1");
// btn1.onclick = () => {
//     console.log("single Click!");
//     alert("Pressed single clicked button");
// };

// //Handle 1
// let btn2 = document.getElementById("btn2");
// btn2.ondblclick = () => {
//     console.log("Double Click!");
//     alert("Pressed double clicked button");
// };

// //Handle 2
// btn2 = document.getElementById("btn2");
// btn2.onmouseover = () => {
//     console.log("Double Click!");
//     alert("Pressed double clicked button");
// };

let modeBtn = document.querySelector("#modeButton");
let body = document.querySelector("body");
let currentMode = "light";

//Way 2
```